



UNIVERSIDADE D  
COIMBRA

Mário Guilherme de Almeida Martins Sequeira Lemos

## NETWORK ANOMALY DETECTION USING IMAGE PROCESSING TECHNIQUES

Dissertation in the context of the Master in Informatics Security, advised by Professor Paulo Simões, Professor Tiago Cruz, Professor Marco Carli, Professor Sara Baldoni and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2025





DEPARTAMENTO DE  
ENGENHARIA INFORMÁTICA  
**FACULDADE DE  
CIÊNCIAS E TECNOLOGIA  
UNIVERSIDADE D  
COIMBRA**

Mário Guilherme de Almeida Martins Sequeira Lemos

# **NETWORK ANOMALY DETECTION USING IMAGE PROCESSING TECHNIQUES**

**Dissertation in the context of the Master in Informatics Security, advised by  
Professor Paulo Simões, Professor Tiago Cruz, Professor Marco Carli,  
Professor Sara Baldoni and presented to the Department of Informatics  
Engineering of the Faculty of Sciences and Technology of the University of  
Coimbra.**

September 2025





DEPARTAMENTO DE  
ENGENHARIA INFORMÁTICA  
**FACULDADE DE  
CIÊNCIAS E TECNOLOGIA  
UNIVERSIDADE DE  
COIMBRA**

Mário Guilherme de Almeida Martins Sequeira Lemos

# **DETEÇÃO DE ANOMALIAS EM REDE UTILIZANDO TÉCNICAS DE PROCESSAMENTO DE IMAGENS**

**Dissertação no âmbito do Mestrado em Segurança informática, orientada pelo  
Professor Paulo Simões, Professor Tiago Cruz, Professor Marco Carli,  
Professor Sara Baldoni e apresentada ao Departamento de Engenharia  
Informática da Faculdade de Ciências e Tecnologia da Universidade de  
Coimbra.**

Setembro 2025



## Acknowledgements

I would like to express my deepest gratitude to the individuals and institutions whose support and guidance have been instrumental in the completion of this dissertation, developed in the context of the Master in Informatics Security with the collaboration of the University of Coimbra and the University of Roma Tre.

First and foremost, I extend my sincere appreciation to my advisors, Professor Tiago Cruz, Professor Paulo Simões, Professor Marco Carli, and Professor Sara Baldoni, for their indispensable guidance, insightful feedback, and constant encouragement throughout this journey. Their input and mentorship were critical in shaping the direction and quality of this work.

I am also profoundly grateful to the Departments of Informatics Engineering of the University of Coimbra and the University of Roma Tre for providing an enriching academic environment and the resources necessary for the successful realization of this dissertation.

Finally, I would like to acknowledge the unwavering support of my family, friends, and colleagues, who supported me throughout this challenging yet rewarding process.

This work is the culmination of collaborative efforts, and I am sincerely thankful to everyone who contributed, directly or indirectly, to its completion.

## Agradecimentos

Gostaria de expressar a minha profunda gratidão às pessoas e instituições cujo apoio e orientação foram fundamentais para a realização desta dissertação, desenvolvida no âmbito do Mestrado em Segurança Informática com a colaboração da Universidade de Coimbra e da Universidade de Roma Tre.

Em primeiro lugar e acima de tudo, estendo o meu sincero agradecimento aos meus orientadores, Professor Tiago Cruz, Professor Paulo Simões, Professor Marco Carli e Professora Sara Baldoni, pela sua orientação indispensável, feedback perspicaz e encorajamento constante ao longo desta jornada. O seu contributo e orientação foram fundamentais para moldar a direção e a qualidade deste trabalho.

Estou também profundamente grato aos Departamentos de Engenharia Informática da Universidade de Coimbra e da Universidade de Roma Tre por terem proporcionado um ambiente académico enriquecedor e os recursos necessários para a realização bem sucedida desta dissertação.

Finalmente, gostaria de agradecer o apoio inabalável da minha família, amigos e colegas, que me apoiaram ao longo deste processo desafiante mas gratificante.

Este trabalho é o culminar de esforços de colaboração, e estou sinceramente grato a todos os que contribuíram, direta ou indiretamente, para a sua conclusão.



## **Abstract**

This research project is dedicated to the development of a network anomaly detection solution, that utilizes image processing techniques. It is intended to establish an intelligent system that can detect cyber-attack instances by analyzing network traffic data. The decision to use a Two-Dimensional (2D) representation derived from the understanding of the efficiency of creating a concise characterization of multi-input information, such as traffic data. By transforming the network flows into images, we can then analyze them as a sequence of frames, allowing for the application of image processing techniques.

The research is focused on two types of cyber attacks, namely Port Scanning and Denial of Service. For the image creation process, a balanced four-channel heatmap approach was utilized, allowing for the retention of spatial and behavioral features of the network. Two different approaches for the detection of attacks were presented: an image-per-attack and a single-image solution. A Convolutional Neural Network (CNN) was utilized for its ability to learn features from images, such as the ones created.

The evaluation of the system was conducted using the UGR'16 dataset, which provided realistic background traffic combined with labeled attack scenarios. The proposed approaches consistently achieved high performance, with classification results reaching approximately 99% in both accuracy and precision across attack categories. While these results demonstrate the effectiveness of the methodology, it is crucial to acknowledge the limitations regarding generalization.

## **Keywords**

Network Anomaly Detection; traffic classification; image processing; 2D data representation; deep learning; real-time, cyber-physical systems.



## Resumo

O presente trabalho de investigação dedica-se ao desenvolvimento de uma solução para a deteção de anomalias em redes, recorrendo a técnicas de processamento de imagem. Pretende-se estabelecer um sistema inteligente capaz de identificar instâncias de ciberataques através da análise de tráfego de rede. A opção pela utilização de uma representação bidimensional advém da compreensão da sua eficiência na criação de uma caracterização concisa de informação de natureza multi-entrada, como é o caso dos dados de tráfego. A transformação dos fluxos de rede em imagens permite a sua análise como uma sequência de frames, possibilitando a aplicação de técnicas de processamento de imagem.

A investigação incidiu sobre dois tipos de ciberataques: Port Scanning e Denial of Service. Para o processo de criação das imagens, foi adotada uma abordagem de mapa de calor de quatro canais equilibrado, permitindo a preservação simultânea de características espaciais e comportamentais da rede. Foram ainda concebidas duas abordagens distintas para a deteção dos ataques: uma baseada na geração de imagens específicas por ataque e outra assente numa representação única. Para a análise das imagens, foi utilizada uma Rede Neural Convolucional, dada a sua reconhecida capacidade de extrair e aprender padrões relevantes a partir de representações visuais.

A avaliação do sistema foi realizada utilizando o conjunto de dados UGR'16, que disponibiliza tráfego de fundo realista combinado com cenários de ataque devidamente etiquetados. As abordagens propostas alcançaram consistentemente um elevado desempenho, obtendo resultados de classificação próximos dos 99% em termos de exatidão e precisão em todas as categorias de ataque. Embora estes resultados evidenciem a eficácia da metodologia, é fundamental reconhecer as limitações existentes no que respeita à capacidade de generalização.

## Palavras-Chave

Deteção de anomalias em rede; classificação de tráfego; processamento de imagens; representação de imagens 2D; aprendizagem profunda; tempo real, sistemas cyber-físicos.



# Contents

|          |                                                           |           |
|----------|-----------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                       | <b>1</b>  |
| 1.1      | Motivation . . . . .                                      | 2         |
| 1.1.1    | Limitations of Rule-Based and Signature-Based Methods . . | 2         |
| 1.1.2    | Call for Advanced Techniques . . . . .                    | 3         |
| 1.1.3    | Leveraging Image Processing and Deep Learning . . . . .   | 3         |
| 1.1.4    | Concrete Attacks . . . . .                                | 3         |
| 1.2      | Potential Challenges . . . . .                            | 3         |
| 1.3      | Structure . . . . .                                       | 4         |
| <b>2</b> | <b>Background Information</b>                             | <b>7</b>  |
| 2.1      | Cyber-Physical Systems . . . . .                          | 7         |
| 2.2      | Network Security . . . . .                                | 7         |
| 2.3      | Traffic Flow . . . . .                                    | 8         |
| 2.4      | Anomaly Detection . . . . .                               | 8         |
| 2.5      | Network Anomaly Detection System . . . . .                | 8         |
| 2.5.1    | Rule-based Systems . . . . .                              | 8         |
| 2.5.2    | Signature-based Systems . . . . .                         | 9         |
| 2.6      | Image Processing Techniques . . . . .                     | 9         |
| 2.7      | Deep Learning . . . . .                                   | 9         |
| 2.7.1    | Evaluating Metrics . . . . .                              | 10        |
| 2.8      | Types of Cyberattacks . . . . .                           | 12        |
| 2.8.1    | Port Scanning . . . . .                                   | 12        |
| 2.8.2    | Denial of Service . . . . .                               | 12        |
| 2.8.3    | Polymorphic Attacks . . . . .                             | 12        |
| 2.8.4    | User to Root Attack . . . . .                             | 13        |
| 2.8.5    | Remote to Local Attack . . . . .                          | 13        |
| 2.8.6    | Probing . . . . .                                         | 13        |
| 2.9      | Summary . . . . .                                         | 14        |
| <b>3</b> | <b>State of the Art</b>                                   | <b>15</b> |
| 3.1      | Datasets . . . . .                                        | 15        |
| 3.1.1    | Reference Datasets . . . . .                              | 16        |
| 3.1.2    | Recent Datasets . . . . .                                 | 21        |
| 3.2      | Traffic Features . . . . .                                | 26        |
| 3.2.1    | General Feature Selection . . . . .                       | 27        |
| 3.2.2    | Features per type of Attack . . . . .                     | 27        |
| 3.2.3    | Denial of Service . . . . .                               | 27        |
| 3.2.4    | Port Scanning . . . . .                                   | 29        |

---

|          |                                         |           |
|----------|-----------------------------------------|-----------|
| 3.3      | 2D Representation Methods . . . . .     | 31        |
| 3.4      | Deep Learning Model . . . . .           | 32        |
| 3.4.1    | Convolutional Neural Networks . . . . . | 33        |
| 3.4.2    | Autoencoders . . . . .                  | 34        |
| 3.4.3    | Hybrid Models . . . . .                 | 35        |
| 3.5      | Complementary Approaches . . . . .      | 35        |
| 3.6      | Summary . . . . .                       | 36        |
| <b>4</b> | <b>Methodology</b>                      | <b>39</b> |
| 4.1      | Planning . . . . .                      | 39        |
| 4.2      | Risks . . . . .                         | 41        |
| 4.3      | Approaches . . . . .                    | 42        |
| 4.4      | Dataset . . . . .                       | 43        |
| 4.5      | Data Handling . . . . .                 | 44        |
| 4.5.1    | Feature Selection Approach . . . . .    | 45        |
| 4.6      | Image Processing Techniques . . . . .   | 46        |
| 4.6.1    | Aggregated Weighted Feature . . . . .   | 47        |
| 4.6.2    | 4-Channel Heatmap . . . . .             | 48        |
| 4.7      | Deep Learning Model . . . . .           | 51        |
| 4.7.1    | Addressing Class Imbalance . . . . .    | 51        |
| 4.8      | Summary . . . . .                       | 52        |
| <b>5</b> | <b>Experiments and Results</b>          | <b>53</b> |
| 5.1      | Single Image Approach . . . . .         | 55        |
| 5.2      | Multiple Image Approach . . . . .       | 58        |
| 5.2.1    | Port Scanning . . . . .                 | 58        |
| 5.2.2    | Denial Of Service . . . . .             | 59        |
| 5.3      | Summary . . . . .                       | 60        |
| <b>6</b> | <b>Conclusion</b>                       | <b>63</b> |
| 6.1      | Future Work . . . . .                   | 64        |
|          | <b>References</b>                       | <b>65</b> |
|          | <b>Appendix A 2D Representations</b>    | <b>71</b> |
| A.1      | Aggregated Weighted Feature . . . . .   | 72        |
| A.2      | 4-Channel Heatmap . . . . .             | 74        |
| A.2.1    | Single Image Approach . . . . .         | 74        |
| A.2.2    | Multiple Image Approach . . . . .       | 78        |

# Acronyms

**2D** Two-Dimensional.

**AUC** Area Under Curve.

**CBAM** Convolutional Block Attention Module.

**CFS** Correlation-based Feature Selection.

**CNN** Convolutional Neural Network.

**CNN-AE** Convolutional Neural Network Autoencoder.

**CPS** Cyber-Physical Systems.

**DDoS** Distributed Denial of Service.

**DoS** Denial of Service.

**DSR** Design Science Research.

**DT** Decision Tree.

**FN** False Negative.

**FP** False Positive.

**FPR/h** False Positives per Hour.

**GA** Genetic Algorithm.

**GAN** Generative Adversarial Networks.

**GRU** Gated Recurring Unit.

**IDS** Intrusion Detection System.

**ISP** Internet Service Provider.

**KNN** K Nearest Neighbor.

**LAN** Local Area Network.

**LSTM** Long short-term memory.

**LSTM-AE** Long short-term memory autoencoder.

---

**NADS** Network Anomaly Detection System.

**PR Curve** Precision-Recall Curve.

**R2L** Remote to Local.

**RF** Random Forest.

**ROC** Receiver Operating Characteristic.

**SVM** Support Vector Machine.

**TN** True Negative.

**TP** True Positive.

**U2R** User to Root.

**VAE** Variational Autoencoders.

# List of Figures

|      |                                                                                                         |    |
|------|---------------------------------------------------------------------------------------------------------|----|
| 3.1  | Example image for a Denial of Service (DoS) attack [Baldoni et al., 2022] . . . . .                     | 32 |
| 3.2  | Classification of deep learning methods for intrusion detection [Chalapathy and Chawla, 2019] . . . . . | 34 |
| 3.3  | Output of the Binvis tool . . . . .                                                                     | 36 |
| 4.1  | GANTT Chart of the work done until the Intermediate Delivery . .                                        | 39 |
| 4.2  | GANTT Chart of the work done until the Final Delivery . . . . .                                         | 40 |
| 4.3  | Approach Diagram . . . . .                                                                              | 42 |
| 4.4  | Topology of the Network from URG16 [Fernández et al., 2017] . . .                                       | 45 |
| 5.1  | Class Distribution for Single Image Approach . . . . .                                                  | 54 |
| 5.2  | Class Distribution for Port Scanning . . . . .                                                          | 54 |
| 5.3  | Class Distribution for Denial of Service . . . . .                                                      | 55 |
| 5.4  | Precision, Recall, and F1 Metrics for Single Image Approach . . . .                                     | 55 |
| 5.5  | Confusion Matrix for Single Image Approach . . . . .                                                    | 56 |
| 5.6  | Skewed Metrics . . . . .                                                                                | 56 |
| 5.7  | Skewed Confusion Matrix . . . . .                                                                       | 57 |
| 5.8  | Confusion Matrix for Port Scanning . . . . .                                                            | 58 |
| 5.9  | Precision, Recall, and F1 Metrics for Port Scanning . . . . .                                           | 58 |
| 5.10 | Confusion Matrix for Denial of Service . . . . .                                                        | 59 |
| 5.11 | Precision, Recall, and F1 Metrics for Denial of Service . . . . .                                       | 60 |
| A.1  | Comparison of Multiple Image Approach for DoS and Port Scanning.                                        | 72 |
| A.2  | Comparison of Single Image Approach Images. . . . .                                                     | 73 |
| A.3  | Benign for Single Image Approach . . . . .                                                              | 74 |
| A.4  | Denial of Service for Single Image Approach . . . . .                                                   | 75 |
| A.5  | Port Scan for Single Image Approach . . . . .                                                           | 76 |
| A.6  | Mean Channel Images per Class for Single Image Approach . . . .                                         | 77 |
| A.7  | Benign for Port Scanning . . . . .                                                                      | 78 |
| A.8  | Attack for Port Scanning . . . . .                                                                      | 79 |
| A.9  | Benign for Denial of Service . . . . .                                                                  | 80 |
| A.10 | Attack for Denial of Service . . . . .                                                                  | 81 |
| A.11 | Mean Channel Images per Class for Port Scanning . . . . .                                               | 82 |
| A.12 | Mean Channel Images per Class for Denial of Service . . . . .                                           | 82 |



# List of Tables

|      |                                                                                                               |    |
|------|---------------------------------------------------------------------------------------------------------------|----|
| 3.1  | Reference Datasets . . . . .                                                                                  | 16 |
| 3.2  | Statistics of Redundant Records in the KDDCUP'99 train dataset<br>[Tavallae et al., 2009] . . . . .           | 19 |
| 3.3  | Statistics of Redundant Records in the KDDCUP'99 test dataset<br>[Tavallae et al., 2009] . . . . .            | 19 |
| 3.4  | Statistics of the randomly selected records from KDDCup'99 train<br>dataset [Tavallae et al., 2009] . . . . . | 20 |
| 3.5  | Recent Datasets . . . . .                                                                                     | 22 |
| 3.6  | Features obtained from the Genetic Algorithm (GA) [Altulaihan<br>et al., 2024] . . . . .                      | 28 |
| 3.7  | Features obtained from the Correlation-based Feature Selection (CFS)<br>[Altulaihan et al., 2024] . . . . .   | 29 |
| 3.8  | Selected Features [Jasim et al., 2023] . . . . .                                                              | 30 |
| 3.9  | Results Obtained from both classification algorithms [Jasim et al.,<br>2023] . . . . .                        | 30 |
| 3.10 | Traffic Features research papers analyzed . . . . .                                                           | 31 |
| 3.11 | 2D Representation Methods research papers analyzed . . . . .                                                  | 32 |
| 3.12 | Deep Learning research papers analyzed . . . . .                                                              | 33 |
| 4.1  | URG16 Anonymized Machine IP Addresses [Fernández et al., 2017]                                                | 44 |
| 5.1  | Comparative Performance of the Different Approaches . . . . .                                                 | 61 |



# Chapter 1

## Introduction

The landscape of Cyber-Physical Systems (CPS) and cybersecurity threats is in continuous evolution, and the need for robust network intrusion detection mechanisms has never been more critical. The current intrusion detection systems struggle in the face of new and adaptive attacks, because of the fact that they mostly rely on rule-based or signature-based methods. These static approaches present a clear limitation, highlighting the necessity for a paradigm shift towards more advanced techniques that can discern subtle anomalies within network traffic data.

This research focuses on developing a network intrusion detection system that attempts to tackle these limitations by presenting a solution based on image processing techniques. The main objective is to create an intelligent system that can discern instances of attacks within any given network by analyzing its traffic data.

Utilizing a 2D representation allows for a more concise form of characterizing multi-input information, such as traffic data. The images created can then be analyzed as a sequence of frames enabling image processing techniques to be used. The final objective will be to define a deep learning model for anomaly detection based on the images created. For the purpose of this research, mainly two types of attacks are going to be considered and analyzed, being Port Scanning and Denial of Service. This decision was made in an attempt to narrow the scope of the work, allowing for a deeper analysis of how the system handles each attack.

This thesis's fundamental objective focuses on defining a network anomaly detection solution for cyber-physical systems based on deep learning techniques and an optimized Two-Dimensional (2D) representation of the network. Multiple approaches will be exposed and analyzed but all will encompass the following steps/structure:

1. selection of the most effective 2D representation of network traffic information;
2. definition of a deep learning model for the detection of anomalies based on the 2D representation of traffic data;
3. design and implementation of a context-sensitive network anomaly detec-

tion system by exploiting the information gathered from a set of distributed nodes.

## 1.1 Motivation

The landscape of CPS and cybersecurity threats is in continuous evolution, and the need for robust network intrusion detection mechanisms has never been more critical. Cyber threats, which can range from subtle intrusions to sophisticated attacks, are continuously challenging the resilience of contemporary network security measures. As CPS blend real-world processes and computational components, they end up playing a central role in different industries that include crucial infrastructures. As these systems continue to integrate diverse subsystems, their attack surface grows accordingly, leading to an increase in the likelihood of previously unknown threats. As these systems become evermore interconnected the effects of successful attacks also increase, which furthermore underlines the constant necessity for intrusion detection solutions [Cruz et al., 2014].

The current intrusion detection systems struggle in the face of new and adaptive attacks, because of the fact that they mostly rely on rule-based or signature-based methods. These static approaches present a clear limitation, highlighting the necessity for a paradigm shift towards more advanced techniques that can discern subtle anomalies within network traffic data. The following sections will delve deeper into the intricacies of these limitations.

### 1.1.1 Limitations of Rule-Based and Signature-Based Methods

These methodologies are known to introduce a high level of rigidity to the intrusion detection system, as their functionality relies on predefined rules. This rigidity means that these systems are only capable of preventing attacks that have been previously studied and analyzed, as they require a new rule or static pattern to be implemented for every new attack vector.

This vulnerability is particularly visible when such systems are faced with polymorphic attacks, as they struggle to adapt to their ever-changing nature. As new and more complex attack methodologies emerge, the limitations of the generalization capabilities of these systems are tested as they will encounter difficulties in the adaptation process.

Another critical issue with these systems is their high-level maintenance needs, as they require an up-to-date signature database to function properly. The constant necessity for updates, for the ability to stay secure against new and evolving threats, not only brings potential gaps in coverage but also impacts the overall effectiveness of the system.

Adding to the problems, the deterministic nature of these systems makes them susceptible to both false negatives and false positives. Any form of deviation from the predefined rules that may occur from normal network behavior can

trigger a false positive warning, increasing its rate. On the opposite side, a slight change or non-explicit anomalies can lead to false negatives, raising questions about the reliability of the system.

### **1.1.2 Call for Advanced Techniques**

Currently, it is becoming clear that the nature of cyber threats is shifting from a static to an adaptive nature, as they have begun to employ sophisticated evasion tactics to avoid static detection mechanisms. Also, the ever-growing volume and complexity of modern networks pose great challenges to the efficacy of the previously mentioned systems.

In the current cybersecurity landscape, the need for real-time responsiveness is imperative. The ability to swiftly detect emerging threats is a fundamental aspect of a strong intrusion detection system. As so, the motivation to create a system that can perform real-time analysis, allowing for quick identification of anomalies and possible attacks.

Deep learning algorithms, more specifically those that perform behavioral analysis, appear as a promising tool for a more dynamic approach. They exhibit a capability to detect anomalies even without any dependency on predefined signatures or rules, which is perfect for a more adaptive and dynamic system, hence the motivation to try and incorporate such a system in this work.

### **1.1.3 Leveraging Image Processing and Deep Learning**

This research is also motivated by the belief that the aggregation of image processing techniques and deep learning can effectively improve the efficiency of anomaly detection in network traffic data. By creating a context-rich characterization of the traffic data it's possible to take full advantage of the benefits of using a deep learning model.

### **1.1.4 Concrete Attacks**

For the testing of an intrusion detection system, it is important to simulate real-world scenarios and attacks. Because of this, it was important to choose the specific attacks that were going to be analyzed. Port Scanning and Denial of Service are common attacks that provide a transparent application of the solution proposed, hence why they were chosen.

## **1.2 Potential Challenges**

As expected, the creation of an Anomaly Detection System comes with a lot of challenges, and some complications are expected to arise. The first issue encoun-

tered is the research of available datasets to be used in the training phase. Even if network security is a well-studied topic, the continuous evolution of attack strategies and communication systems leads to a proliferation of methods to guarantee their security. Unfortunately, the availability of verification datasets does not follow the same updating trend. Furthermore, the use of deep learning-based methods requires a large amount of data to effectively train networks. Consequently, the choice of the dataset containing the traffic data to be analyzed is an important step towards the realization of the proposed anomaly detection system. To this aim, two key aspects must be considered: the sampling interval and the total duration of the data recording. The sampling interval must be short and fixed for all recorded data. If the sampling interval is long, assuming that the recorded data is analyzed by exploiting time windows, to collect enough samples in each of them, a single time window would correspond to a long period, thus impacting the system's promptness. On the contrary, the use of smaller time windows would result in the processing of a reduced number of samples, thus impairing the system's effectiveness.

As for the total data recording period, it should be long enough for both normal and anomalous traffic to perform effective training and testing.

Moreover, one of the first problems to deal with is the pre-processing of the network data and the definition of its structure so that the effectiveness of a subsequent analysis module may be maximized. More specifically, the following characteristics need to be selected:

1. the traffic parameters (e.g., bytes, packets) to be used in the two-dimensional representation of the network status;
2. the data normalization model to guarantee that the resulting images have the same dynamic range;
3. the best domain representation of the 2D data. Some methods might be too complex to apply in a real scenario, while others may be too hard to interpret;
4. the type of information that needs to be represented for each traffic parameter (e.g., the traffic volume, the correlation between traffic patterns of different nodes). The main problem in choosing the parameters and their weights is that some might make it easier to detect an attack, and others a different type of attack.

## 1.3 Structure

This paper is structured around six main chapters, with each addressing a key component of the research process and methodology employed for the development of a Network Anomaly Detection System (NADS) using image processing techniques. A brief overview of each chapter is provided next.

- Chapter 1 - Introduction

This chapter provides a general overview of the research problem, the motivations behind it, and the objectives pursued. It also presents the main challenges of the current NADSs and the possible complications that may appear when applying the proposed methodology.

- Chapter 2 - Background

This chapter is related to the introduction and explanation of all the basic information that is going to be mentioned throughout this thesis, including concepts of CPS, anomaly detection, and deep learning.

- Chapter 3 - State of the Art

All of the research that was conducted regarding the topics and respective subtopics is presented in this chapter, alongside a discussion about the main alternatives that could have been selected. It references existing work on datasets, traffic features, 2D representations, and deep learning models for the purpose of intrusion detection.

- Chapter 4 - Methodology

Details regarding the Design Science Research (DSR) approach are present in this chapter, along with details about the decision-making process for the selection of the dataset, image construction methods, and the Convolutional Neural Network (CNN) model design.

- Chapter 5 - Experiments and Results

Presents the implementation details and evaluation metrics used to assess the NADSs performance. The results obtained for the multiple approaches are analyzed and discussed.

- Chapter 6 - Conclusion

Provides a summary of the findings and conclusions, highlighting the contributions and also providing context on the limitations of the work, along with future work possibilities.



# Chapter 2

## Background Information

In this chapter, a small introduction to important concepts will be presented. This allows for the establishment of a foundational understanding of multiple key terms and considerations that are necessary for a better understanding of the network anomaly detection system later described. It is important to mention that the concepts addressed will be explained within the context of this thesis, and not in their complete magnitude, in order to maintain this description short and within scope.

### 2.1 Cyber-Physical Systems

A Cyber-Physical Systems (CPS) is a computer system where a mechanism is controlled by computer-based algorithms. It contains software and physical components and is usually designed as a network of interacting elements. Because of their interconnect ability and reliance on networked communication, they introduce vulnerabilities, making intrusion detection mechanisms very important in the protection against potential threats.

Some examples of CPSs are smart grids, industrial control systems, and autonomous automobile systems.

### 2.2 Network Security

Networks are constantly being targeted by continuously evolving attacks. New and sophisticated methods are continually being created in an attempt to compromise the confidentiality, integrity, and availability of networked systems. Because of this, network security tries to secure it by applying multiple techniques, such as adaptive anomaly detection systems.

## 2.3 Traffic Flow

In a Network, a flow can be defined as one or more packets that travel between two computer addresses utilizing a specific protocol (TCP, UDP, ICMP, etc.) and a particular set of ports that are defined for each end of the flow. Every packet contains this tuple of information (host source/destination, port source/destination, protocol). For communication in the opposite direction, the source and destination elements that are present in the tuple will be reversed, this means that a flow can be either half-duplex or full-duplex. [Moore et al., 2005]

## 2.4 Anomaly Detection

Given the context of network security, anomaly detection is the process of identifying deviations from what is considered normal behavior. It plays a crucial role in this context, as the identification of unusual patterns or behaviors may indicate security threats. There are many methods for anomaly detection, such as signature-based, behavioral-based, and machine learning-based.

Signature-based methods struggle to adapt to evolving threats and can only detect previously documented attacks. Behavioral-based methods try to learn what is the normal behavior of a network to detect deviations, but it needs a learning period to establish the normal behavior and it generates false positives. Machine Learning-based methods use machine learning algorithms to identify anomalies and can adapt to new patterns of attack, the main issue is that it requires a significant amount of training data.

## 2.5 Network Anomaly Detection System

A Network Anomaly Detection System (NADS) is a fundamental cybersecurity tool that meticulously analyzes and identifies abnormal or suspicious traffic patterns in a network environment. It works on the premise that normal network behavior possesses certain predictable patterns, which allows the establishment of the expected behavior of a network under normal conditions.

As both rule-based and signature-based NADSs are going to be mentioned later, a brief description of both is presented next.

### 2.5.1 Rule-based Systems

A rule-based NADS operates through predefined rules or policies that can be defined and customized by network security administrators. By analyzing the network traffic against the predefined rules, it is able to flag traffic that violates them, classifying it as a potential threat.

Their main advantage is the customization ability that allows them to be effective in identifying deviations from the predefined rules.

Their main limitation is the adaptability power against evolving threats or anomalies that are not covered by the currently defined rules.

## **2.5.2 Signature-based Systems**

A signature-based NADS relies on a database of already known patterns or signatures that are associated with what can be considered normal network behavior. They function by comparing the network traffic against some predefined signatures and, similarly to rule-based systems, flag deviations as possible anomalies.

In such a system, the dataset should be continuously updated to include new signatures that can reflect the discovery of new threats. They can be effective in identifying an anomaly pattern, but face a limitation when detecting a new anomaly as it will not match against the existing signatures.

## **2.6 Image Processing Techniques**

Image processing techniques allow for the transformation of raw network traffic data into a 2D representation. This is useful as it provides a visual abstraction from the extremely complex information collected from network probes. This allows for a more context-rich and intuitive characterization of a network, which will directly improve the accuracy of the anomaly detection system.

## **2.7 Deep Learning**

Machine Learning is a very broad topic, and deep learning is a subset of it. It involves training neural networks with multiple layers to automatically learn hierarchical representations of data. It works especially well to detect intricate patterns and relationships within large datasets. This characteristic is extremely useful for analyzing traffic data, hence why its adaptability and scalability are exploited in an attempt to improve the accuracy of the anomaly detection system.

Even though there are different deep learning architectures, Convolutional Neural Networks (CNNs) stand out for the NADS presented in this thesis. CNNs can exploit the spatial structure present in images through the use of convolutional layers, which apply learnable filters across the input image. Each filter acts as a detector for local patterns; in this case, it's combinations of traffic feature intensities. This is particularly advantageous for this scenario, as attacks such as Denial of Service (DoS) or Port Scanning can often leave distinctive visual signatures in a 2D representation of flows. For example, a DoS attack can appear as a concentration of unusually high intensities in a certain region of an image, and a Port Scan

can display a systematic pattern across multiple ports. Of course this all depends on how the 2D representation is created, but it presents a good advantage.

### 2.7.1 Evaluating Metrics

In order to evaluate the performance of a predictive model, multiple metrics can be used. Some of the most common ones are accuracy, precision, recall, F1 score, ROC and AUC, Precision-Recall Curve (PR Curve), and False Positives per Hour (FPR/h). Each metric provides a different perspective on its performance. These are calculated through the use of the True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) parameters. These represent:

- TP: number of instances correctly predicted as positive.
- FP: number of instances incorrectly predicted as positive.
- TN: number of instances correctly predicted as negative.
- FN: number of instances incorrectly predicted as negative.

#### Accuracy

Accuracy is the ratio of correctly predicted instances to the total instances in the dataset, measuring the percentage of times the model was correct overall. The formula to calculate it is presented below.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

#### Precision

Precision is the ratio of correctly positively predicted instances to the total of positive predictions. This metric is specifically important for applications where having false positives leads to a high cost. Its formula is presented below.

$$\text{Precision} = \frac{TP}{TP + FP}$$

#### Recall

Recall is the ratio of correctly positively predicted instances to all instances of the same class. It fundamentally highlights the number of times the model correctly predicted positive out of all the positive instances, as it can be understood from its formula. Contrary to precision, this metric becomes even more important when the cost of a false negative is high.

$$\text{Recall} = \frac{TP}{TP + FN}$$

### **F1 Score**

The F1 score provides an overall score as it combines both precision and recall, as it can be seen in the formula below. This metric allows for the observation of the amount of false positives and false negatives simultaneously.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

### **ROC and AUC**

The Receiver Operating Characteristic (ROC) curve is a graphical representation of a model's performance across all the decision thresholds. It is constructed by plotting the True Positive Rate (TPR) on the y-axis against the False Positive Rate (FPR) on the x-axis. A perfect classifier would have a ROC curve that passes through the top left corner of the graph.

$$\text{False Positive Rate} = \frac{FP}{FP + TN}$$

The Area Under Curve (AUC) of a ROC curve is a numeric value ranging from 0 to 1 that summarizes the ROC curve, and it can be interpreted as the probability that a randomly chosen positive instance is correctly ranked higher than a randomly chosen negative instance.

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR}) d\text{FPR}$$

An AUC of exactly 1 means that the classifier is perfect, while an AUC of 0.5 is equivalent to random guessing.

### **Precision-Recall Curve**

In cases where the dataset is extremely imbalanced, meaning that attacks are much rarer than benign traffic, ROC curves can present a more optimistic view of the performance of a system. In those cases, PR Curves can be more assertive, as it plots precision against recall; meaning that if the area under the PR Curve is high the system can detect attacks with high recall while also keeping false positives low.

## **False Positive per Hour**

In a real-world scenario, this metric is one of the most important as it relates to the false positive rate normalized to time. Unlike some of the previous abstract ratios exposed, this metric relates directly to an analyst's workload in a real system, meaning that if there are many false positives generated per hour the NADS is unlikely to be practical, even if the recall is fairly high. This metric allows us to understand how deployable a system is.

## **2.8 Types of Cyberattacks**

It's impossible to discuss intrusion detection systems without mentioning cyberattacks, so the following topics will be all regarding different types of attacks mentioned throughout the paper.

### **2.8.1 Port Scanning**

Port Scanning is a technique that can be used to discover the open ports on a computer or networked device. A port is a communication endpoint that facilitates and processes the sharing of information between different applications or services. It can serve multiple purposes, such as security assessments, network troubleshooting, or cyberattacks. [Kaushik et al., 2010]

It revolves around the principle of sending a series of connection requests or small messages, to the target device's ports and analyzing the response queue to see if they are closed, open, or filtered by firewalls. Some of the most common scans are TCP, SYN, UDP, and Stealth scans. [Kaushik et al., 2010]

### **2.8.2 Denial of Service**

A DoS attack is a cybersecurity threat whose primary goal is to disrupt the normal functioning of a network, by making it unavailable to users. Its primary objective is to overwhelm the target's resources and it can be achieved by, for example, flooding it with an enormous amount of requests or traffic. This will tamper with the target's ability to handle requests, causing a disruption to the availability of the services. [Elleithy et al., 2006]

### **2.8.3 Polymorphic Attacks**

A polymorphic attack refers to a type of attack characterized by its ability to dynamically alter its code or characteristics while maintaining its core malicious functionality. Its name is derived from the concept of polymorphism, where a single entity can manifest multiple forms. [Chauhan and Shah Heydari, 2020]

Their primary objective is to evade traditional security detection mechanisms, attempting to bypass signature-based detection. Techniques such as behavioral analysis and machine learning can be employed to increase the capacity of intrusion anomaly detection systems against these types of attacks. [Chauhan and Shah Heydari, 2020]

#### **2.8.4 User to Root Attack**

A User to Root (U2R) attack is a type of cyber attack where an unauthorized user who possesses limited privileges attempts to escalate them and gain the highest level of access to a network or system. [Jeya et al., 2012]

This attack takes advantage of security weaknesses in the system and can allow the attacker to gain control over critical components.

#### **2.8.5 Remote to Local Attack**

A Remote to Local (R2L) attack is a type of cyberattack where an attacker, who is located outside of the targeted system's local network, attempts to gain access and control over a system that is inside the target's local network. They contrast attacks that originate within the same network or system, as they involve an external entity trying to compromise the system from a distance. [Jeya et al., 2012]

They try to exploit vulnerabilities in network protocols, configuration settings, or software to access the system remotely. After gaining access, the attacker can try to exploit weaknesses in the system to gain control.

The implementation of intrusion anomaly detection and prevention systems can allow for the detection and mitigation of these types of attacks.

#### **2.8.6 Probing**

A probing cyberattack, also known as network probing or reconnaissance, is a method utilized by attackers to learn information regarding the targeted system or network. Its primary goal is to identify possible vulnerabilities or weaknesses and understand what security measures are in place. [Ahmad et al., 2009]

It's considered an initial step of a cyberattack, as it is used to gain information to launch better-targeted attacks next.

The network anomaly detection system detailed in this paper will attempt to detect a method of probing called Port Scanning.

## 2.9 Summary

This chapter provides an introduction to multiple important concepts relevant to NADSs. The first topics approached are basic notions regarding the main concept, followed by the definition of NADS and their different relevant types. A definition of deep learning and important subtypes is also present, followed by a description of multiple relevant cyberattacks.

In this chapter, we established a basic understanding of multiple concepts critical to the development and comprehension of a NADSs. We first explored CPSs, emphasizing their vulnerabilities and the necessity for robust intrusion detection mechanisms. This was followed by an examination of network security and the ongoing battle against cyberattacks. An extensive definition of network traffic flow and its bifurcated communication pathways provided a necessary backdrop for understanding the subsequent discussion on anomaly detection.

# Chapter 3

## State of the Art

As the cybersecurity landscape has evolved, the necessity for more adaptive and robust intrusion detection systems has been highlighted and has become evident. Because of this, some research has already been performed regarding these systems, and many different approaches can be found.

This chapter is an examination of the current state-of-the-art methodologies, focusing mainly on the key points and decisions that must be taken to create the proposed system. Different research papers were analyzed specifically for each topic. In this way, a better understanding of the current advances, methodologies, and challenges associated with each topic is presented, allowing for a deeper understanding of how the methodology detailed in Chapter 4 was achieved.

For a better understanding of the overall subject, this chapter will be divided into five subsections. Firstly, an introduction to all the past and currently used datasets will be presented, as understanding their characteristics is essential. Subsections 3.2, 3.3, and 3.4 present the choices and approaches made by multiple different papers regarding traffic feature selection, Two-Dimensional (2D) representation methodologies, and deep learning models, respectively. Lastly, subsection 3.5 presents an overview of some more complementary approaches.

### 3.1 Datasets

The concept of Network Anomaly Detection Systems (NADSs) has been around for a long time now, and ever since, efforts for the creation of datasets that allow their evaluation and testing have been made. However, the lack of public datasets for NADSs is highlighted in multiple research papers. [Tavallaee et al., 2009] [Fernández et al., 2017] Regarding the first built datasets, they are called reference datasets, as they have been used in a large number of research papers. [Fernández et al., 2017] To better show the main characteristics of these datasets, Table 3.1 on page 16 is presented.

It's important to establish that, because of the nature of the subject, only datasets that include labeled data (labeled as in it representing an attack instance or not).

Non-labeled attacks aren't of much use to evaluate NADS.

| Reference                | Dataset    | Time Span | Connection Records | Format |
|--------------------------|------------|-----------|--------------------|--------|
| [Lab, 1998a]             | DARPA'98   | 7 weeks   | 5M                 | pcap   |
| [Lab, 1998b]             | DARPA'99   | 5 weeks   | -                  | pcap   |
| [S., 1998]               | KDDCup'99  | 7 weeks   | 2M                 | csv    |
| [Tavallaee et al., 2009] | NSL-KDD    | 7 weeks   | 125,973            | csv    |
| [Perona et al., 2008]    | gureKDDCup | 7 weeks   | -                  | csv    |

Table 3.1: Reference Datasets

As it can be deduced through the nomenclature of the presented datasets, the first effort for the creation of such datasets were the DARPA datasets, carried out by the MIT Lincoln Lab. [Fernández et al., 2017] Intrusion Detection Systems (IDSs) were tested in a off-line evaluation using network traffic and audit logs collected on a simulation network. The data was processed in batch mode by the systems as they attempted to identify attack sessions that were mixed with normal network activities. Both of these datasets were evaluated both in off-line and in real-time. [Lab, 1998a]

### 3.1.1 Reference Datasets

#### DARPA'98

DARPA'98 contains training data, that spans 7 weeks of synthetic network traffic containing 5M connection records, and testing data that spans 2 weeks and contains 2M connection records. [Fernández et al., 2017]

The main technical objectives were to evaluate the ability of an IDS to detect in any of the following attacks events were attempted or occurred during a network session:

1. Denial of Service
2. Unauthorized access from a remote machine
3. Unauthorized access to local superuser privileges by a local unprivileged user
4. Surveillance and Probing

During the 7 weeks 174 attack instances were performed from 27 different attacks (all deriving from the 4 main presented above; some were performed in multiple sessions/instances). Some anomalous user behaviour was also introduced during the 6th week. These included users logging in at unusual hours or with different identities. [Lab, 1998a]

The results obtained were considered extremely successful, has the first large baseline corpus of background traffic and attacks for algorithm development and testing was created. The evaluation performed showed that in the best combination, around 65% of attacks were detected. There was a noticeable difference in the ability to detect new versus old attacks, this was especially noticeable in Denial of Service (DoS) and Remote to Local (R2L) attacks, with around 20% and 10% new attacks detected versus around 80% old attacks detected. [Lab, 1998a]. It's important to note that these results are heavily influenced by the IDS used at the time and aren't necessarily a reflection of the quality of the dataset.

**Main Problems:** In an attempt to protect privacy, both the benign and attack data were synthesized. It is claimed that the data is similar to the one sampled in multiple similar conditions during a period of several months, even though no analytical or experimental validation tests were performed on the data's false alarm characteristics. Also, the workload the data presents doesn't seem comparable to the traffic of a real-world network. [Tavallaee et al., 2009]

DARPA'98 does not contain a definition of the network attacks, as in what is considered one. As an example, not all packets that cause a buffer overflow should be considered as an attack instance. Another case would be that probing isn't necessarily considered an attack until the number of interactions it performs breaches a certain threshold. [Tavallaee et al., 2009]

## DARPA'99

Similar to the '98 edition, DARPA'99 also contains training data and testing data. In total, there are 5 weeks of tcpdump network traffic files that were sniffed from a synthetic network from two points, one inside the network and one outside. The host base audit data from 4 different victim systems were also collected. [Fernández et al., 2017]

Regarding the training data, it spans 3 weeks, but it as the particularity of the first and last weeks containing no attack instances. This decision was made to facilitate the training of NADs. Only the second week contains attack instances that are a subset of the DARPA'1998 attacks, plus several new ones. [Lab, 1998b]

As for the testing data, it is comprised of two weeks of network-based attacks mixed with normal background data. It contains 201 instances of 56 different types of attacks. [Lab, 1998b]

There were 17 different types of DoS attacks, 12 types of User to Root (U2R) attacks, 15 types of R2L attacks, and 10 types of probing attacks. [Lab, 1998b]

It's important to mention that traffic collectors like `tcpdump`, the one used in this dataset, possess a known issue that was not addressed. They have a high chance of overloading and dropping packets when a heavy traffic load is present. Nothing was done to check if such a situation occurred.[Tavallaee et al., 2009]

### KDD Cup 1999

This dataset was first used in the Third International Knowledge Discovery and Data Mining Tools Competition (better known as KDDCup'99), which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition mainly focused on the building of intrusion detection systems, having the main objective of encouraging the development of effective intrusion detection systems. [S., 1998]

This dataset was built based on the DARPA'98 and contains around 2M connection records and 4,9M single connection vectors, each one with 41 features and labeled as being part of an attack instance (with the type of attack explicitly defined) or normal. Its comprised of 7 weeks of raw TCP dump data, that was generated in a simulated environment of a military network: including both normal and intrusive activities. The various types of attacks present were intended to simulate different types of real-world cybersecurity threats, namely DoS, U2R, R2L, and Probing. Based on these four main attack types, 24 attack types are included in the training dataset and an additional 14 in the test dataset only, in an attempt to improve the realism of the test data. [Fernández et al., 2017] Regarding DoS neptune, land, back, pod, smurf, and teardrop attacks were introduced; for U2R buffer overflow, rootkit, perl, and loadmodule; as for R2L ftp write, guess password, imap, multihop, phf, spy, warezclient, and warezmaster; lastly regarding Probing ipsweep, nmap, satan, and portsweep attacks were conducted. [S., 1998]

The features present in the dataset fit into 3 major groups: basic features, traffic features, and content features. The basic features encapsulate all the attributes the can be gathered from a TCP/IP connection. These features mostly delay to an implicit delay in detection. The traffic features encapsulate all features that are achieved by considering a time interval, and are split into two groups: same host features and same service features. They both examine the behaviour of a connection in the last 2 seconds that either have the same destination host or same service at the current connection, respectively. There is also a special category called connection-based traffic features; it was creation because certain attacks, such as slow probing attacks, don't produce a intrusion pattern inside the 2 second window. To combat this issue, the same host and same service features are re-calculated for a connection window of 100 connections. Lastly, content features are related to the data portion of the packets (e.g. number of failed login attempts). These exists for the detection of U2R and R2L attacks, has they are embedded in the data portion of the packet and usually only require one connection. [Tavallaee et al., 2009]

**Main Problems:** The main problem related to this dataset is the extremely high value of redundant records. This makes all the learning algorithms biased to the frequent records, making it less capable of detecting the ones that are infrequent. The fact that the test dataset also contains a high number of redundant records leads to a biased evaluation of the methods that better detect the frequent records. [Tavallaee et al., 2009] For a better understanding of the relevancy of this problem, Tables 3.2 and 3.3 were included.

| -       | Original Records | Distinct Records | Reduction Rate |
|---------|------------------|------------------|----------------|
| Attacks | 3,925,650        | 262,178          | 93.32%         |
| Normal  | 972,781          | 812,814          | 16.44%         |
| Total   | 4,898,431        | 1,074,992        | 78.05%         |

Table 3.2: Statistics of Redundant Records in the KDDCUP'99 train dataset [Tavallaee et al., 2009]

| -       | Original Records | Distinct Records | Reduction Rate |
|---------|------------------|------------------|----------------|
| Attacks | 250,436          | 29,378           | 88.26%         |
| Normal  | 60,591           | 47,911           | 20.92%         |
| Total   | 311,027          | 77,289           | 75.15%         |

Table 3.3: Statistics of Redundant Records in the KDDCUP'99 test dataset [Tavallaee et al., 2009]

Another obvious problem is the fact that this dataset was developed using attack patterns that were in use in the 90s and that can be outdated by now. The network also represented a less complex scenario in today's standard.

## NSL-KDD

This dataset is an extension and enhancement of the previously mentioned dataset KDDCup'99, and it was created to deal with some of the problems described earlier. The types of attacks that are present are unchanged and an extra feature has been added. The best way to understand what was done is by comparing it to its predecessor.

## Main Differences between KDD Cup 1999 and NSL-KDD

As stated above, one of the main problems of the KDDCup'99 dataset is the number of redundant records. In an attempt to combat this problem, all of the du-

plicated records were removed, keeping only distinct records. Tables 3.2 and 3.3 indicate the reduction rate obtained during this process. There were also two invalid records found during this process that ended up being removed. [Tavallae et al., 2009]

The usual approach that was taken when using the KDDCup'99 dataset to train a machine learning algorithm was studied by [Tavallae et al., 2009]. An experiment that entailed training seven widely used machine learning techniques using their default input values was performed to check the degree of difficulty of getting high accuracy percentages. Three subsets of the main train dataset were randomly selected and used on all seven learners, creating 21 learning machines. From these tests, a total of 97.97% and 86.64% of the train and test dataset respectively have been correctly labeled by all the classifiers. This exposes that evaluating methods using the KDDCup'99 dataset based on accuracy, false positive rate, and detection rate is not and appropriate solution. In an attempt to increase the degree of difficulty a new train and test dataset were created. [Tavallae et al., 2009]

| -            | Distinct Records | Percentage | Selected Records |
|--------------|------------------|------------|------------------|
| 0-5          | 407              | 0.04       | 407              |
| 6-10         | 768              | 0.07       | 767              |
| 11-15        | 6,525            | 0.61       | 6,485            |
| 16-20        | 58,995           | 5.49       | 55,757           |
| 21           | 1,008,297        | 93.80      | 62,557           |
| <b>Total</b> | 1,074,992        | 100.00     | 125,973          |

Table 3.4: Statistics of the randomly selected records from KDDCup'99 train dataset [Tavallae et al., 2009]

Table 3.4 show the number and percentage of records that were correctly labeled by a certain number of learned machines. It attests to the previous statement that an extremely high percentage of records was correctly labeled in all 21 instances. To create the new datasets, the records chosen were randomly selected from each value group in a inversely proportional percentage. [Tavallae et al., 2009] As an example, a percentage of 5.49 of the records are present in the 16-20 value group, and so, 94.51 of these records were selected. This totaled in 125,973 and 22,544 records being selected respectively for the new training and test datasets. [Tavallae et al., 2009]

After performing the same test with the newly created dataset, the overall distribution of the percentage of records correctly labeled changed significantly. The percentage of records correctly labeled in all 21 instances decreased around 10%. This discrepancy shows that the original dataset is unproportionately distributed making it not suitable for testing NADS. [Tavallae et al., 2009]

It's also important to refer that by presenting a dataset with a significantly smaller size, there is no longer a need to select a subset to perform the experiments. This also allows for the results of the evaluation performed by different research works to be comparable.

### **gureKDDCup**

Similarly to NSL-KDD dataset, gureKDDCup was created based in the KDD-Cup'99 in an attempt to improve it. It also bases itself in the predecessor of all of the previously mentioned datasets, DARPA'98. [Perona et al., 2008]

The main objective in the creation of this dataset was to add payload information to the KDDCup'99 dataset. This means that basically, gureKDDCup is the KDDCup'99 dataset plus the payload and basic information such as IP address and port number for each connection. [Perona et al., 2008]

This means that this new dataset will share all the main problems pointed out before at 3.1.1.

### **Summary**

Now that these datasets characteristics and origins have been exposed, the data presented at Table 3.1 can be better understood. The datasets time spans are mostly the same because they all somewhat derived from the DARPA'98. All of them have been deeply analyzed and their flaws exposed in multiple research papers. Even though they were once the standard, nowadays they are not capable of providing a good evaluation for the methods tested on them, and so a search for more recent datasets should be made.

### **3.1.2 Recent Datasets**

Many of the datasets exposed in the following sections were created because their authors noticed a lack of suitable ones, as stated in [Sperotto et al., 2009], [Wang et al., 2017], and many others.

There are many challenges that explain this lack of datasets. According to [Sperotto et al., 2009], one of the main difficulties is achieving a satisfactory balance between privacy and realism. The most realistic traces are, of course, captured in a real network such as corporate networks or Internet service providers. The issue is that these traces may reveal private information regarding the involved entities. In contrast artificial traces avoid this problem but usually require a deeper domain knowledge to achieve a realistic result. Because of this, many research papers utilize non-public traffic traces. The usage of self-collected or private traffic damages the credibility of the results obtained [Wang et al., 2017]. Additionally, labelling is an extremely time consuming process, specially considering that a high number of traces is important for a satisfactory evaluation [Sperotto et al., 2009].

Regarding the requirements that are ideally present in a dataset, [Fernández et al., 2017] presented the following: features, real background traffic, updated attack traffic, labeling, duration, documentation, and format. The dataset should present the appropriate features for the type of NADS that is going to be evaluated, or present the data in a raw format allowing for the researches to extract the features they desire. Regarding the duration, it should ideally span through multiple cycles of study enabling the consideration of cyclostationary evolution of traffic during day/night and weekdays/weekends. Datasets usually present their information in either pcap (tcpdump), csv or flow (netflow) format. Regarding the pcap format, it usually includes the payload information, which can present privacy concerns because anonymizing it reliable is complicated, but allows for a more comprehensive evaluation of NADS. pcap also creates a huge volume of data for datasets with a long duration. In a csv format, data has been preprocessed, containing an extraction of relevant records for certain features. The flow format is usually preferred when dealing with real traffic.

Table 3.5 presents the more recent datasets that were found together with some of their characteristics. Similarly to the reference datasets, they will be further analyzed in the following sections.

| Reference                 | Dataset          | Time Span                          | Connection Records | Format        |
|---------------------------|------------------|------------------------------------|--------------------|---------------|
| [Sperotto et al., 2009]   | Sperotto         | 6 days                             | 14.2M              | flow          |
| [Fontugne et al., 2010]   | MAWI Working Lab | 9 years (15 min/day)               | -                  | pcap          |
| [Shiravi et al., 2012]    | UNB ISCX         | 7 days                             | 2.45M              | csv           |
| [García et al., 2014]     | CTU-13           | 142 hours                          | 81M                | pcap and flow |
| [Creech and Hu, 2013]     | ADFA-LD12        | -                                  | 5400               | csv           |
| [Moustafa and Slay, 2015] | UNSW-NB15        | 31 hours                           | 2M                 | csv           |
| [Wang et al., 2017]       | USTC-TFC2016     | traffic captured between 2011-2015 | 752,040            | -             |
| [Fernández et al., 2017]  | UGR'16           | 134 days                           | 16.9M              | flow          |

Table 3.5: Recent Datasets

## Sperotto

In the research paper where this dataset was introduced [Sperotto et al., 2009], the author highlighted in several occasions that there weren't any datasets that complied with all the specifications they required. These are that it should be realistic, completely labeled, contain the attack types of interest, be publicly available and, for their specific application, be flow-based traffic traces.

The presents the first labeled dataset for flow-based intrusion detection, containing 14.2M flows. The data was gathered throughout a period of 6 days, that was claimed as sufficient by the authors as it contained weekday and weekends instances. [Sperotto et al., 2009] This would be true if an extensive cyclostationarity evaluation of a network behaviour wasn't of interest, but networks present not only short-term but also long-term temporal behaviour. [Fernández et al., 2017]

After studying and presenting multiple methods for capturing real network traffic, in order to guaranty a satisfactory confidence level and ability to label the traffic in a reasonable time-frame, the topology of the network became very simplistic. Only a single host with enhanced logging capabilities was monitored (honeypot). The traffic was labeled through the use of the registered logs, meaning that it depends on the detection system configuration and the log processing. They claimed to have been able to correctly label more than 98% of the traces.

The dataset isn't completely available to the public, but in a on-request system through e-mail to the authors.

## MAWI Working Lab

The dataset presented in the [Fontugne et al., 2010] research paper, is a labeled version of the MAWI (Measurement and Analysis on the WIDE Internet) archive sample points B and F. It is comprised of 15 minutes daily traffic traces captured from a link between Japan and the United States. Even though the data is publicly available, the payloads are omitted and the IP addresses are anonymized. The traffic began being captured in 2001, but only the first week of every month until 2009 was used to evaluate the similarity estimator. The whole 9 years was only used later for testing the final combinations.

Four unsupervised anomaly detectors that are based on statistical-analysis techniques were implemented. Their results were compared through a similarity estimator. The dataset was labeled using the SCANN combination strategy and given one of four different labels: Benign, Notice, Suspicious, and Anomalous.

The fact that this dataset is extremely extensive, allows for a analysis of a very long period, but the trace being of only 15 minutes a day becomes a limitation. The labelling is also dependant on the classifiers that have been used in the process. [Fernández et al., 2017]

### UNB ISCX

This dataset was presented in the [Shiravi et al., 2012] research paper and introduced the concept of profiles. A profile attempts to facilitate the reproduction of real-world behaviours has seen from a network by containing an abstract representation of certain features and events. Two main profiles are defined  $\alpha$  and  $\beta$  representing profiles for attack traffic and background traffic, respectively. The data collection spanned across 7 days and resulted in a total of 2,450,324 flows.

The topology of the network used is comprised of 20 Windows XP stations and one Windows 7 computer, divided into 4 Local Area Networks (LANs) to effectively represent a interconnected network.

The main drawbacks of this dataset nowadays is the use of synthetic traffic, the usage of old operating systems and a somewhat reduced duration. [Fernández et al., 2017]

### CTU-13

The authors of this research paper [García et al., 2014], realized the lack of satisfactory datasets that contained botnet attack traffic, along with a methodology to compare different methods for its detection. Even though this dataset is focused only in the detection of botnet attacks, it is still relevant to mentioned it because of its efforts to capture real traffic.

The topology included a group of virtualized computers running Windows XP that were part of the botnet and were being bridged to a University network. The traffic was captured through a single University router and the previously mentioned machines. Since all the IP addresses of the infected botnet machines were known, these were used for the labeling process. The traffic captured in the University routes was used as the background and normal traffic.

Thirteen different botnet scenarios were created, with each one representing of a different malware behaviour. In total, the data was captured during around 142 hours and 81M flows were captured. On average, throughout the 13 scenarios, the amount of background traffic was around 91%, around 6% of normal traffic and around 3% of botnet traffic. The authors attempted and were successful in creating this traffic distribution to accurately represent a real scenario where a botnet attack traffic only represents a small percentage of the overall network traffic.

For privacy reasons, the original pcap files were not made publicly available, but the labeled NetFlow files that were used to test the methodologies were. One of the issues with this dataset is that it doesn't allow for the consideration of cyclostationary behaviour because of its reduced duration.

### ADFA-LD12

This dataset that was presented in [Creech and Hu, 2013], focuses on Host-based intrusion detection systems. The main objective of this research paper was to

prove that the reference datasets, more specifically the KDDCup'99, can no longer be used to accurately evaluate NADSs and create a new dataset. In an attempt to strike an effective balance between data fidelity and processing burden, a low amount of traces were captured. For the training data 833 traces were collected; regarding the validation data 4373 traces and 60 attack instances were performed.

When comparing the results obtained from 4 well-know methodologies against the KDDCup'99 and the newly presented dataset, the initial affirmations made were supported. The methods performed considerably worst with the new dataset, and the conclusion taken by the authors was that their dataset provided a considerably higher difficulty level, specially when compared to the almost guaranteed good results obtained from the reference dataset.

The main drawback of this dataset is the lack of description regarding the single Linux OS machine where the traces associated with normal traffic is collected. This impossibilities the generalization between what is considered normal behaviour to the authors and other machines in the Internet. [Fernández et al., 2017]

### **UNSW-NB15**

The UNSW-NB15 dataset was proposed by [Moustafa and Slay, 2015]. A synthetic environment with three virtual servers was configured using the IXIA traffic generator. An extra two routers completed the topology of the simulated network. The tcpdump tool was utilized to capture the Pcap files of the simulation that spanned a total of 31 hours, gathering around 2M flows. From these files, 49 features were extracted using the Argus and Bro-IDS tools together with twelve algorithms.

Regarding the labelling step, the report generated about the attack data from the IXIA tool is used. Each record is labelled with an attack category and a 0 or 1 (for normal or attack respectively). There are a total of nine categories of attacks: fuzzers, analysis, backdoor, DoS, exploits, generic, reconnaissance, shellcode, and worms. The distribution of the traffic is of around 13% malicious and 87% normal. [Moustafa and Slay, 2015]

The main drawback of this dataset is its synthetic environment and traffic generation, as it is associated with theoretical and not realistic network behaviours. [Fernández et al., 2017]

### **USTC-TFC2016**

The authors of the [Wang et al., 2017] research paper decided to created their own dataset after highlighting the lack of suitable ones. As their approach included using a Convolutional Neural Network (CNN) for representation learning, it made sense utilizing a dataset that contained the raw traffic data, as being able to learn features directly from it is one of its greatest advantages.

The raw traffic dataset is divided into two parts, one containing malware traffic,

and the other normal traffic. The malware traffic is divided into ten types and was collected between 2011 and 2015 by CTU researchers from public websites that were in a real network environment. The normal traffic is also divided into ten types that were collected through the use of the IXIA BPS tool. There is a total of 752,040 records in the entire dataset.

### UGR'16

This dataset present in [Fernández et al., 2017] contains two main subsets. The first contains only real background traffic data gathered during four months (March to June 2016 totaling 100 days), and the second one contains real background data (collected from July to August 2016 totaling 34 days) and synthetically generated that correlates with multiple types of cyberattacks.

The data was obtained from a Tier 3 Internet Service Provider (ISP) real network. This allows the dataset to represent real-world traffic generated by internet users, especially when compared to others created in a controlled environment where only certain traffic present appears.[Casarin et al., 2022]

The attacks that were synthetically added corresponded to DoS, Port Scanning, and some Botnet traffic was also added. These attacks were added using planned and random scheduling simultaneously. [Baldoni et al., 2022]

## 3.2 Traffic Features

From the previous section 3.1, we understand that there are multiple ways the traffic data can be presented in a dataset. Some present the raw data, while others already give us the features they consider more useful directly. This allows for, depending on the dataset chosen, different approach to be applied regarding the handling of the traffic data. As mentioned above, each dataset possesses a multitude of different features that can be useful for an IDS. The ones used must be carefully picked if we want to achieve a high percentage of detection, as they are the metrics to understand if an attack is ongoing or not. Without a significant difference between how these features describe an attack instance from all others the overall approach will never produces successful results.

As it is expected, not all features carry the same descriptive power over the network. Some are critical to understand its state, while others can be used for fine-tuning the IDS. A balance between these must be found for the system to work as expected. The performance of a feature-based detection will depend on the number of features and their representativeness. Selecting a low number may lead to an inaccurate detection, while too many will increase the overhead. The representativeness of a feature relates to the variance between its selected state and its typical value [Yue et al., 2020].

After an initial analysis of the literature, an important question was raised, as there appeared to be two possible ways of choosing the best features. Choosing

global features that must be able to allow the detection of all the types of attacks tested or choose specific features for each attack type. Multiple approaches are going to be detailed in the following sections.

### 3.2.1 General Feature Selection

In reviewing the selected approach, it became evident that a significant number of research papers tended to identify and utilize a similar set of features. This is mainly because the importance of certain features is undeniable.

### 3.2.2 Features per type of Attack

This presents a trade-off between adding more complexity to the system but also increasing the efficacy of the IDS. But firstly, to understand what features to choose, an understanding of the attacks that are to be detected is essential; a description of each attack is present in Section 2.8.

### 3.2.3 Denial of Service

Sudden spikes in the overall network, or an unusually high rate of incoming packets can indicate a DoS attack. Certain DoS attacks exhibit distinctive packet size characteristics, and so a large number of either small or large packets may be indicative. The same goes for a large number of abnormally long or short connections.

DoS attacks usually involve a flood of incoming packets, which alters the regular pattern of packet flows, pointing to unusual activity.

For this type of attack, features like packet rate, traffic volume, packet size, packet direction, connection duration, and minimal time between two flows are relevant and can be enough to detect it.

The paper [Altulaihan et al., 2024] focuses on the detection of DoS attacks. Even though it doesn't utilize any 2D representation methodologies, it does an extensive research on the best features to detect such an attack. They utilized two different feature selection algorithms, the Correlation-based Feature Selection (CFS) algorithm and the Genetic Algorithm (GA), and compared the results obtained. The following tables present the features selected by each algorithm. To give more insight into the performance of the algorithms, the tests were also executed using all the available features without any previous selection.

The GA selected 13 out of the original 80 as the most relevant, while the CFS algorithm organized them from most relevant to least, the top 13 was selected for a fair comparison between both. As it can be observed from the results, only one feature was selected by both algorithms, the Flow\_IAT\_Mean (mean time between two packets sent in the flow).

| Feature Name    | Description                                                                                      |
|-----------------|--------------------------------------------------------------------------------------------------|
| Src_Port        | Source Port Number                                                                               |
| Dst_Port        | Destination Port Number                                                                          |
| Tot_Fwd_Pkts    | Total packets in the forward direction                                                           |
| Tot_Bwd_Pkts    | Total packets in the backward direction                                                          |
| Fwd_Pkt_Len_Max | The maximum size of the packet in the forward direction                                          |
| Flow_IAT_Mean   | Mean time between two packets sent in the forward direction                                      |
| Bwd_IAT_Std     | Standard deviation time between two packets sent in the backward direction                       |
| Bwd_URG_Flags   | Number of times the URG flag was set in packets travelling in the backward direction (0 for UDP) |
| SYN_Flag_Cnt    | Number of packets with SYN                                                                       |
| URG_Flag_Cnt    | Number of packets with URG                                                                       |
| Down/Up_Ratio   | Download and upload ratio                                                                        |
| Idle_Mean       | Mean time a flow was idle before becoming active                                                 |
| Idle_Max        | The maximum time a flow was idle before becoming active                                          |

Table 3.6: Features obtained from the GA [Altulaihan et al., 2024]

Once the features were selected, they were tested using four different classifier algorithms, Decision Tree (DT), Random Forest (RF), K Nearest Neighbor (KNN), and Support Vector Machine (SVM). They were evaluated using the accuracy, precision, recall, and F1 score metrics. These metrics were computed using the True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) parameters.

All three approaches performed extremely well under the metrics presented, revealing values over 99% for all the classifier algorithms; with the exception of the SVM combined with the GA that presented values over 91%. The best results were obtained with the features selected by the GA combined with the RF and DT classifiers. Even though it wasn't large, there was still a noticeable improvement when using the feature selection algorithms.

| Feature Name    | Description                                                        |
|-----------------|--------------------------------------------------------------------|
| Idle_Min        | The minimum time a flow was idle before becoming active            |
| Bwd_Header_Len  | Total bytes used for headers in the backward direction             |
| Fwd_IAT_Mean    | Mean time between two packets sent in the flow                     |
| TotLen_Fwd_Pkts | The total size of the packet in forward direction                  |
| Fwd_Pkt_Len_Max | The maximum size of the packet in forward direction                |
| Fwd_Pkts/s      | Number of forward packets per second                               |
| Flow_IAT_Std    | Number of forward packets per second                               |
| Pkt_Len_Var     | Variance length of a packet                                        |
| Idle_Std        | Standard deviation time a flow was idle before becoming active     |
| Fwd_IAT_Max     | The maximum time between two packets sent in the forward direction |
| Pkt_Len_Max     | The maximum length of a packet                                     |
| Flow_IAT_Min    | The minimum time between two packets sent in the flow              |
| Flow_IAT_Mean   | Mean time between two packets sent in the forward direction        |

Table 3.7: Features obtained from the CFS [Altulaihan et al., 2024]

### 3.2.4 Port Scanning

Because of the fact that Port Scanning often entails a systematic exploration of a system by systematically probing multiple ports, a unique and distinctive connection rate pattern usually appears.

Frequent and sequential connections (or attempts) to different ports from a single source are major indicators of a possible attack. The connections are usually short as the attack's purpose is reconnaissance. The same logic applies to the size of the packets.

For this attack, some of the relevant features are source and destination ports, connection duration, packet size, and retransmission rate.

The paper [Jasim et al., 2023] proposed a method of detecting Port Scanning attacks using a classification algorithm. They do not make use of the advantages

of a 2D representation but perform multiple techniques of feature selection. This allows for a deeper knowledge of the best features to detect Port Scans.

The best features were selected through the use of the information gain and variance techniques. They are then ranked according to their weight and the authors decided to discard all that had a value below 0.6. The features selected and their corresponding weight can be seen in the table

| FeatureName             | FeatureScore |
|-------------------------|--------------|
| TotalLengthofBwdPackets | 0.939343     |
| BwdPacketLengthMax      | 0.939343     |
| BwdPacketLengthMin      | 0.80995      |
| BwdPacketLengthMean     | 0.782456     |
| MaxPacketLength         | 0.782456     |
| PacketLengthMean        | 0.781841     |
| PSHFlagCount            | 0.781841     |
| AveragePacketSize       | 0.778016     |
| AvgBwdSegmentSize       | 0.77582      |
| SubflowBwdBytes         | 0.760317     |
| Init_Win_bytes_backward | 0.708411     |
| act_data_pkt_fwd        | 0.706064     |
| min_seg_size_forward    | 0.706064     |

Table 3.8: Selected Features [Jasim et al., 2023]

Once the features were selected, two different classification algorithms were tested, a Decision Table and a OneR based classifier. They were evaluated using the usual metrics, as it can be seen in the table 3.9. The values indicate that the features selected were in fact capable of providing enough context for the classifier model, as they all exceed 99%.

| Algorithm      | Accuracy | Recall | Precision | F-Measure |
|----------------|----------|--------|-----------|-----------|
| Decision table | 99.82%   | 99.70% | 99.90%    | 99.80%    |
| OneR           | 99.57%   | 99.40% | 99.60%    | 99.50%    |

Table 3.9: Results Obtained from both classification algorithms [Jasim et al., 2023]

Some of the features selected in the analyzed papers are indicated in Table 3.10, alongside the types of attacks that were present in the raw data used.

| Reference              | Type of Attack | Features Selected                                     |
|------------------------|----------------|-------------------------------------------------------|
| [So and Reddy, 2005]   | DoS            | source/destination addresses and port, traffic volume |
| [Baldoni et al., 2022] | Port Scanning  | packet size, connection duration, number of flows     |
| [Baldoni et al., 2022] | DoS            | volume of traffic and its standard deviation          |
| [Moore and Zuev, 2005] | -              | flow Duration, port, size, inter-arrival time         |
| [Taheri et al., 2018]  | Botnet         | representation learning                               |

Table 3.10: Traffic Features research papers analyzed

### 3.3 2D Representation Methods

The advantages of utilizing a 2D representation for the purpose proposed in this work have already been noted, and so some research was found regarding it. The two main steps where multiple solutions can be found are how to index the rows and columns and how to weigh the values for each feature in order to calculate each pixel's intensity.

One of the methods found related the rows of the image to the source IPs and the columns to the destination IPs. At first, this seemed like a solid way of transforming the context of the network into a 2D environment, as it would allow us to even understand where the attack originated from and its specific target. The main problem with this approach was that you can't control the number of source IPs, making the number of rows not controllable. This will also lead to a very sparse image; it would mostly be a black image with a few small white dots that would represent the possible attacks. A way to possibly make this effect less noticeable would be to group the IPs into defined ranges. [Baldoni et al., 2022]

The intensity of the pixels is calculated having in mind the following logic steps:

- firstly checking for multiple flows exchanged with the attacker, this is done by deleting isolated flows.
- because the packets are usually of the same size, the standard deviation of size should be close to zero for the packets sent by the attacker. Because of this, low standard deviation values are valued.

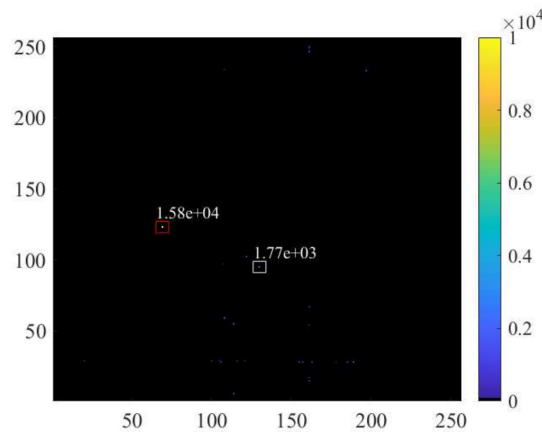


Figure 3.1: Example image for a DoS attack [Baldoni et al., 2022]

- lastly, since with the criteria above, some normal traffic could still be falsely detected, the number of flows will be valued following an exponential function.

Table 3.4 presents more concisely some information regarding the methodology used in each major research paper analyzed.

| Reference              | Methodology           |
|------------------------|-----------------------|
| [So and Reddy, 2005]   | DCT with 32*32 blocks |
| [Baldoni et al., 2022] | Detailed above        |
| [Taheri et al., 2018]  | -                     |
| [Casarin et al., 2022] | -                     |

Table 3.11: 2D Representation Methods research papers analyzed

## 3.4 Deep Learning Model

Deep learning has become one of the most promising paradigms for anomaly detection in recent years. Unlike traditional machine learning, which often depends on manual feature engineering, deep neural networks are capable of automatically learning hierarchical feature representations from data. This ability is particularly valuable in the domain of intrusion detection, where traffic patterns are complex, high-dimensional, and may exhibit subtle differences between benign and malicious activity.

Although multiple architectures have been proposed, as it can be seen in Table 3.5, where we expose the methods utilized in every paper that was researched,

most works in network anomaly detection leverage CNNs, Autoencoders, or hybrid combinations with recurrent models such as Long short-term memorys (LSTMs) and Gated Recurring Units (GRUs). Every one of these approaches provides different advantages depending on whether the focus is spatial representation, reconstruction-based anomaly scoring, or temporal modeling.

| Reference                       | Method                                                                   |
|---------------------------------|--------------------------------------------------------------------------|
| [Zhang et al., 2018]            | MSCRED                                                                   |
| [Kwon et al., 2019]             | Fully Connected Network                                                  |
| [Hartpence and Kwasinski, 2020] | Sequential Neural Networks                                               |
| [Pang et al., 2021]             | GAN, Distance-based measure, Clustering-based Measure                    |
| [Chalapathy and Chawla, 2019]   | DNN, STN, SPN, Generative Models, CNN, Sequence Models, and Autoencoders |
| [Haque et al., 2022]            | Learning-based anomaly detection                                         |
| [Taheri et al., 2018]           | CNN                                                                      |
| [Wang et al., 2017]             | CNN                                                                      |
| [Casarin et al., 2022]          | VAE                                                                      |

Table 3.12: Deep Learning research papers analyzed

Something that was made clear from [Haque et al., 2022] was the necessity of determining the anomaly threshold automatically and adaptively, alongside the benchmarking with sufficient real-world anomalous data.

### 3.4.1 Convolutional Neural Networks

CNN, are the popular choice of neural networks for analyzing visual imagery [Krizhevsky et al., 2012]. CNNs ability to extract complex hidden features from high-dimensional data with complex structure has enabled its use as a feature extractor in outlier detection for both sequential and image datasets [Gorokhov et al., 2017]. They are able to exploit local spatial correlations, due to the use of convolutional layers that apply filters across the input image, enabling the detection of low-level patterns such as edges or pixel intensity clusters, which in the context of network traffic can correspond to attack patterns. Through the use of successive layers, CNNs learn increasingly abstract representations, ranging from local flow features to global traffic structures.

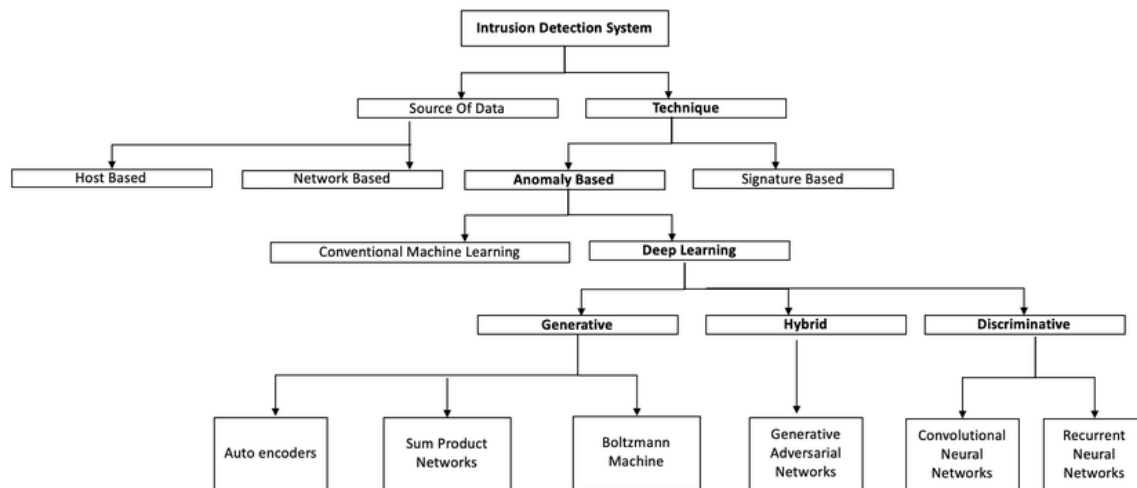


Figure 3.2: Classification of deep learning methods for intrusion detection [Chalapathy and Chawla, 2019]

CNNs also utilizes pooling layers, which reduce the dimensionality of intermediate feature maps while preserving essential information, leading to an improvement in its robustness to minor variations in the input. After several convolutional and pooling stages, the learned features are passed through fully connected layers that act as classifiers, translating them into probabilistic outputs. This hierarchical feature learning allows CNNs to generalize well even when they are trained on less-than-ideal traffic datasets.

CNNs can work directly with raw traffic data [Wang et al., 2017] and, since the route we are taking is of selecting features and processing the data before making the 2D transformation, the full capabilities of CNN can be utilized without extensive manual feature engineering.

### 3.4.2 Autoencoders

Autoencoders represent data within multiple hidden layers by reconstructing the input data, effectively learning an identity function. A problem with them is that when trained mostly on normal data instances (which are a major part of anomaly detection applications), fail to reconstruct the anomalous data samples, therefore, producing a large reconstruction error. Several variants of autoencoder architectures exist to combat this aspect. The selection of an autoencoder architecture is contingent upon the characteristics of the data that is going to be used. Convolutional networks are preferred for image datasets, while LSTM based models work better with sequential data. Using combined models such as Convolutional Neural Network Autoencoders (CNN-AEs) or Long short-term memory autoencoders (LSTM-AEs) once again eliminates the need for the entire data processing and feature selection process.

### 3.4.3 Hybrid Models

Beyond the previously mentioned models, some recent approaches attempt to employ a hybrid deep learning model combining multiple strengths. A CNN-RNN hybrid is particularly prominent, as CNN layers can extract spatial features while recurrent layers such as LSTMs or GRUs can capture temporal correlations between consecutive windows [Cao et al., 2022].

In the [Cao et al., 2022] paper, a CNN-GRU model is proposed, creating a model with the following structure: firstly, the features are assigned different weights with the use of a Convolutional Block Attention Module (CBAM); secondly, through a CNN module, the spatial features are extracted and their information aggregated using both Averagepooling and Maxpooling; then the temporal features are extracted by multiple GRU units, and lastly the classification is carried out with the use of the Softmax function.

Generative models such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) have also been explored; [Casarin et al., 2022] and [Pang et al., 2021] respectively. These models aimed to learn the distribution of benign traffic and flag the deviations as anomalies.

A VAE learns a distribution over latent codes, supporting two natural anomaly signals: reconstruction error, and approximate likelihood or reconstruction probability under the learned generative model. By feeding the VAE with the created images that represented normal network traffic, the reconstruction error was utilized to detect the presence of anomalies [Casarin et al., 2022]. Contrasting a plain AE that can under-penalize subtle deviations in sparse images, the KL-Divergence of a VAE improves generalization to benign variation, enabling likelihood-based scoring which can be more useful if the benign traffic is extremely diverse.

GANs utilize a generator and a discriminator, and, in the case of anomaly detection, the proposition is to make the generator learn how to generate only benign-like data, so that real anomalous samples can be flagged by their incompatibility with the generator-discriminator pair. GANs often generate more realistic reconstructions than VAEs, produce useful anomaly-sensitive features, and work well in unsupervised settings where anomalies are rare and labels scarce; they also have some training instabilities and their anomaly score definition varies, making evaluation less standardized [Pang et al., 2021].

## 3.5 Complementary Approaches

The [Bendiab et al., 2020] paper proposed a similar goal to ours but for IoT applications. They created their own dataset, composed of normal traffic captured from the Cyber-trust project network and malicious traffic gathered from multiple public sources (malware traffic analysis repository, NETRESEC repository, and the malware dataset from the stratosphere lab). All traffic was captured using tools like Nmap and Wireshark from real networks. The malicious traffic includes

various types of attacks, such as DoS, OS scans, Key loggers, and spyware. This was all stored in PCAP files.

After collecting all the traffic, they divided it into multiple packet chunks in order for it to be ready for the image creation process. For the creation of the images, they utilized the binvis [Cortesi, 2015] tool. This tool allows for a visualization of binary data with the ability to cluster bytes to pick out structural features using space-filling curves. They feed the tool with the PCAP files that were previously stored in binary format. In total, 1000 images of normal and malware traffic were collected and utilized for the training of the model. Each image has a size of 784 bytes.

The model used was the Residual Neural Network (ResNet50), which is a pre-trained deep learning model. It is pre-trained against millions of images and utilizes a CNN, making it really useful in this application. The created images are then classified using this model, and this information is stored in a database. In the testing phase, the traffic is compared with the information in the database to get a classification. Any new malware traffic detected will be sent straight in the training dataset, in an effort to improve the model.

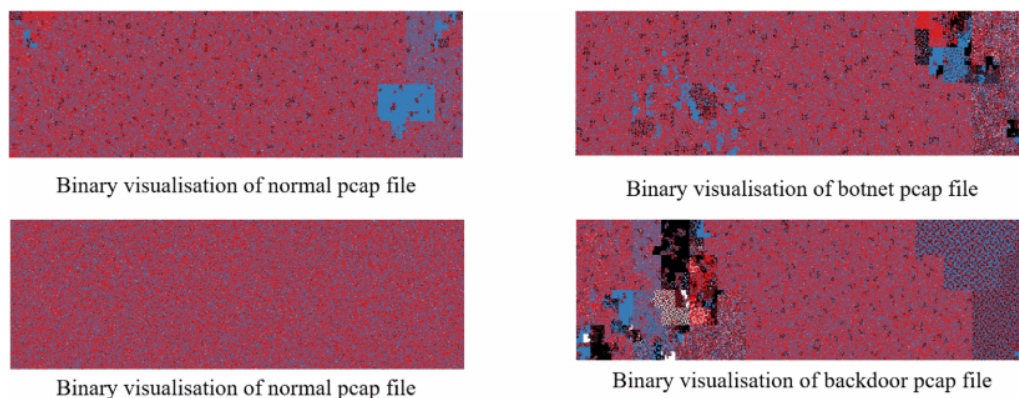


Figure 3.3: Output of the Binvis tool

The results achieved were very positive, as the classifier achieved an accuracy of 94.50% and a precision of 95.78%. Using the same images and process, some other models were tested. These were the SOINN, Mobilenet, and the same Residual Neural Network, but with 34 layers instead of the original 50. They all performed around 3% worse compared to the ResNet50 in all the recorded metrics.

Our goal was always to achieve a more customized solution, to the point of defining the weight of a given feature in the image creation step. This meant that the use of this or similar tools wouldn't be adequate for the project; nevertheless, it's important to highlight these approaches.

## 3.6 Summary

In this chapter, an examination of the current state-of-the-art methodologies was presented, focusing on the key topics and considerations for creating an effective

NADS. It's divided into different sections, namely dataset selection, traffic feature selection, 2D representation methods, and deep learning models.

Regarding the dataset, multiple were analyzed, more specifically KDD CUP 1999, NSL-KDD, and URG'16, describing their characteristics, advantages, and disadvantages. The challenges of selecting a dataset were also highlighted.

For the feature selection section, multiple examples of the features selected by different research papers for DoS and Port Scanning attacks were presented, alongside the reasoning behind such choices.

Different methods for representing network data in a 2D format were exposed. A special focus was given to the indexation of the rows and columns, and the calculation of the intensity of the pixels.

Also, different deep learning models and approaches were analyzed with detail, allowing us to understand their intrinsic value to these applications. CNNs were utilized in multiple approaches with promising results, making it a prime model to possibly utilize in our approach.

This chapter serves as a foundation for understanding the existing landscape of NADS research, setting the stage for the proposed methodology in Chapter 4.



# Chapter 4

## Methodology

### 4.1 Planning

Up to now, the work performed has been more focused on understanding and researching the topic and possible datasets, methods, and the different decisions that had to be made.

The following image represents a GANTT chart, making it easy to visualize what has been done and how it was organized. It also contains information on how the rest of the work is going to be organized.

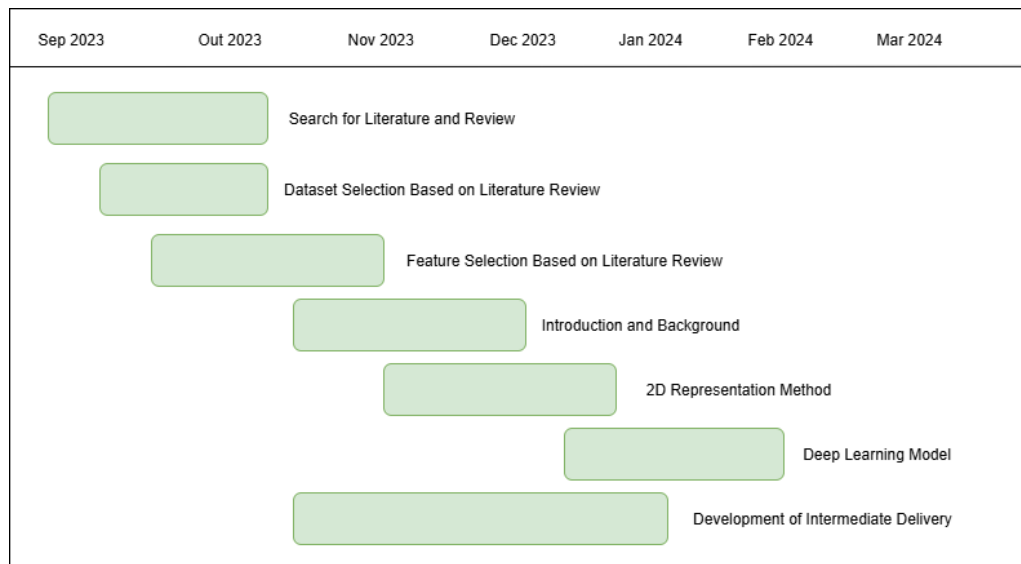


Figure 4.1: GANTT Chart of the work done until the Intermediate Delivery

After approaching a lot of different methods and analyzing different approaches to the topic in question in the last chapter, it is important to outline the proposed methodology for our work. So, in this chapter, all the decisions made using all the previous knowledge as the basis will be pointed out and explained.

Also, all the steps that must be taken on the practical side of the work are presented, constructing the overall process and methodology that will be set in place

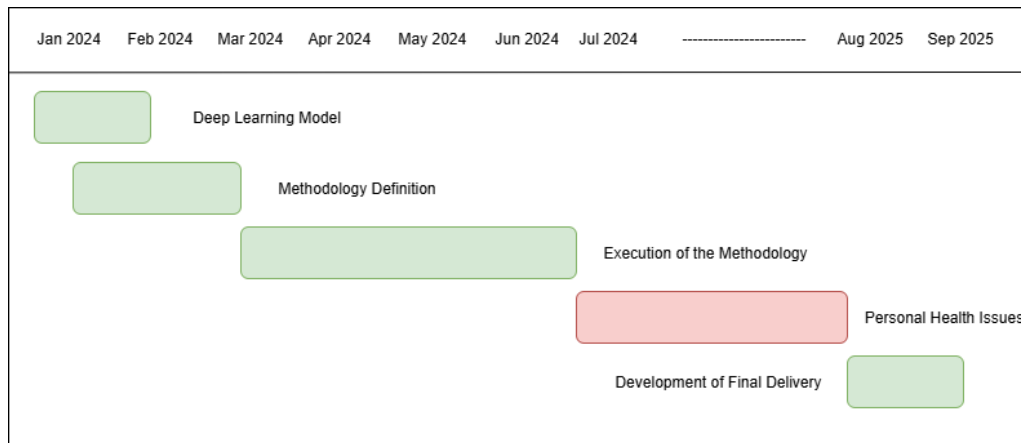


Figure 4.2: GANTT Chart of the work done until the Final Delivery

for the creation of our proposed intrusion detection system.

Regarding the research methodology applied, a Design Science Research (DSR) approach was used, as it aligns with the problem-solving nature of the research objectives. To briefly explain what DSR is, it's a research paradigm that focuses more on the creation and development of innovative solutions to real-world problems. This contrasts with traditional scientific research, which aims to understand and explain concepts.

The main characteristics of DSR are:

- Problem-Solving Orientation

DSR is driven by a practical problem or a need for improvement in a specific area. Researchers aim to design and implement solutions that are effective in addressing these problems.

- Artifact Creation

The primary outcome of DSR is the creation of artifacts, in this case, a Network Anomaly Detection System (NADS), that attempts to solve the problem of detecting Port Scanning and Denial of Service (DoS) attacks.

- Iterative Process

It often involves an iterative process of design, development, evaluation, and refinement. To achieve a satisfactory result, the solution presented should be tested and adjusted iteratively.

- Communication of Design Knowledge

Not only the final artifact but also the knowledge that was gained throughout the process of creating it should be documented. This allows for a more insightful understanding of the decisions that were taken in order to reach the final solution.

## 4.2 Risks

Multiple risks will be present during the development of this methodology, and some of them are going to be presented now, together with a possible mitigation plan.

### Dataset Representation and Diversity

- Risk - The dataset may not be representative enough and not be able to capture the full diversity of real-world scenarios. The UGR'16 dataset, while extensive, is still limited to specific periods, protocols, and artificially introduced attacks. An inadequate representation could bias the model toward specific traffic patterns, reducing its effectiveness when applied to different environments.
- Mitigation - Careful selection of representative datasets, by evaluating its distribution across time, protocols, and attack types. The dataset must possess the criteria stated in section 3.1 to make sure the deep learning model can be trained effectively. Within UGR'16, ensure balanced sampling across weeks and sub-networks.

### Class Imbalance

- Risk - The dataset presents a strong imbalance between benign and attack flows, which directly translates to a much larger number of benign images when comparing to DoS or Port Scanning images. This imbalance can make a model become biased towards predicting the majority class, as this results in high accuracy, but also poor recall.
- Mitigation - In order to address such class imbalance, techniques such as under-sampling of benign traffic, oversampling of attack instances, or the use of class weighting in the Convolutional Neural Network (CNN) loss function can be applied.

### Resource Intensiveness

- Risk - Implementing analysis of network parameters, image processing, and deep learning models may impose a resource burden on the system. The entire pipeline involves heavy operations: parsing millions of flows, transforming them into images, and training deep neural networks. This can create bottlenecks in memory usage, disk I/O, and GPU/CPU training time. If this is not optimized, the system might be impractical for real-time detection or large-scale development.
- Mitigation - An optimization of all the different steps of the system must be performed, together with the balancing of its resources and the employment of scalable infrastructures. This can mean optimizing the preprocessing by

working in chunks (avoiding loading entire files into memory) while also taking full advantage of efficient libraries. Regarding the model training, mechanisms like early stopping and batch normalization can help reduce resource consumption. Experimenting with smaller image resolutions or lightweight architectures is a possible path that allows for the discovery of the trade-offs between accuracy and efficiency.

### False Positives and Negatives

- Risk - Inaccurate detection may lead to operational issues. False positives, flagging legitimate traffic as malicious, lead to unnecessary alerts that overwhelm analysts, and false negatives, which allow actual attacks to go unnoticed. Both reduce trust in the NADS.
- Mitigation - Calibration of the system through an iterative process can help reduce these problems. This can mean the iterative calibration of the thresholds for classification decisions, especially when evaluating multi-class scenarios, and also applying iterative fine-tuning with cross-validation. The usage of metrics like Precision-Recall curve, Receiver Operating Characteristic (ROC) curve, and False Positives per Hour (FPR/h) is important to understand a system's performance and deployability.

## 4.3 Approaches

After a lot of reflection on how to tackle the problem at hand, we decided to experiment with two different approaches. In the overall picture, both paths are similar as the fundamental usage of a 2D representation and a deep learning model is shared.

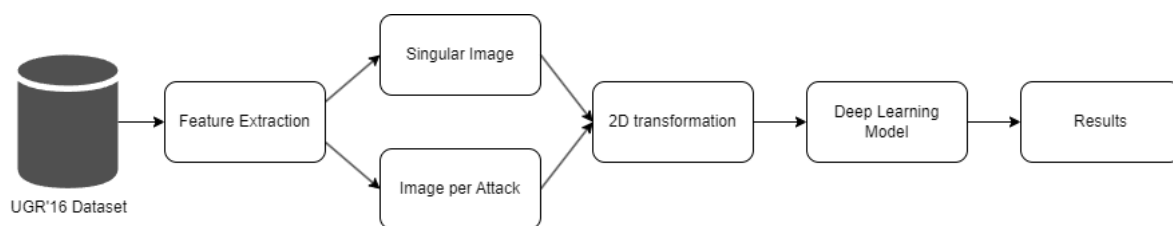


Figure 4.3: Approach Diagram

As can be seen in the Figure 4.3, the two approaches vary in a structural element, but the flow of operations is the same. The bottom path is based on an Image-per-Attack methodology; in this approach, a different image is created for each attack. This narrows the scope of the NADS but, in exchange, will be very beneficial, not only too analyze any possible differences in efficiency but also to further understand the most effective features for each attack using the DSR methodology. In contrast, the top path is based on the selection and manipulation of features in order to detect both a DoS and a Port Scanning attack while using a singular image. From the comparison of the results obtained from these two

approaches, a relation between the performance of creating more/fewer images and the results obtained in each can be created. They were developed in this order because knowing which features work best for each attack will allow us to understand which ones are common between them, as they must be utilized in the top approach.

## 4.4 Dataset

Since the objective is to utilize a deep learning model, the dataset used must be chosen having in mind certain important characteristics. It needs to have a large amount of data, possess the two types of attacks that are going to be studied (port scanning and DoS), and have a satisfying window of attack time [Baldoni et al., 2022]. Because of these restrictions and the previously highlighted lack of useful datasets, not many options were found [Wang et al., 2017].

After studying the existing datasets, one of them distinguished himself from the others by meeting all of the main requirements. A suitable dataset was found in UGR'16. Some of the details of this dataset were already presented in 3.1.2 but, since it was selected, a deeper analysis of its organization is presented next.

As previously stated, the dataset is composed of two big groups of data, the calibration and the test. The traffic data used was the one present in the Test set and that spanned 6 weeks. The data is presented in weekly chunks, each with different attacks present. For each week, there are multiple files available: the entire traffic flows in two different presentations (binary netflow in nfcapd format and CSV netflow flows that are labeled), a CSV file with all the timestamps (in minutes) of all the attacks executions, and multiple CSV files that contain only the flows of each type of attack.

Both DoS and Port Scanning are only present in week 5 of July, week 1 of August, and week 2 of August. Between the three weeks, there are around 9.2 million flows labeled DoS and around 5.4 million labeled Port Scans. This means that a clear class imbalance is present, since there are hundreds of millions of benign flows compared to approximately fifteen million attack flows. This imbalance must be addressed during both the image creation and training steps, and is further explained in Subsection 4.7.1.

This organization helped a lot in the process of bringing all the flows of DoS and Port Scanning together in two separate groups. Having the data organized in this way will be helpful in future procedures.

The nfcapd format includes all the netflow features, while the CSV file is preprocessed and only includes the most frequently utilized features. Each row of the CSV files has thirteen columns that represent: the time of the end of a flow, the duration of a flow, source IP address, the destination IP address, the source port, the destination port, the protocol, the flags, the forwarding status, the type of service, the packets exchanged in the flow, size (in bytes) of the packets exchanged, and the label [Fernández et al., 2017]. Multiple labels were used, one for each

type of attack and one for flows that represent regular traffic. It is very important to mention that the authors did not account for possible attacks that an unknown entity might have performed on the network, which might cause an error in the labeling of the benign flows.

In the Figure 4.4, we can see the topology of the network where the data was extracted. As previously stated, the network is of an ISP and is composed of two subnetworks, the core network and the inner network. The first is where services not protected by a firewall are served, and the second is already protected by a firewall. The attacker network is deployed at the top level and is composed of five attackers. In the core network, five victim machines were placed but used only for the dataset collection. Lastly, on the inner network, an additional fifteen victim machines were placed in three different and already existing networks. On the Table 4.1, it is possible to see the anonymized IP addresses of each machine. Having this information is very valuable, especially since an image is supposed to represent only a single network.

| Attacker/Victim Machines | IP Addresses                     |
|--------------------------|----------------------------------|
| A1 - A5                  | 42.219.150.{246,247,243,242,241} |
| V11 - V15                | 42.219.156.{30,31,29,28,27}      |
| V21 - V25                | 42.219.158.{16,17,18,19,21}      |
| V31 - V35                | 42.219.152.{20,21,22,23,18}      |
| V41 - V45                | 42.219.154.{69,68,70,71,66}      |

Table 4.1: URG16 Anonymized Machine IP Addresses [Fernández et al., 2017]

## 4.5 Data Handling

The CSV file has already been preprocessed and labeled. Since each CSV file correlates to the data of an entire week, in order to read a file, the data needed to be split. Chunks of 70 million lines were used. The data must also be separated between the multiple networks presented; otherwise, the transformation to a 2D representation using the processing technique that was developed and detailed in section 4.6 cannot be performed. The 4.1 table can be consulted to check the four groups of IP addresses that are present in the four networks of victim machines. Since each row/flow has a value that represents a timestamp, the data for each network can easily be grouped into intervals. An interval of one second was utilized, aligning with literature recommendations for balancing temporal granularity and responsiveness.

To ensure a smooth process in the creation of the images, it is important to clean the dataset. This meant checking for any null values or malformed rows and removing them.

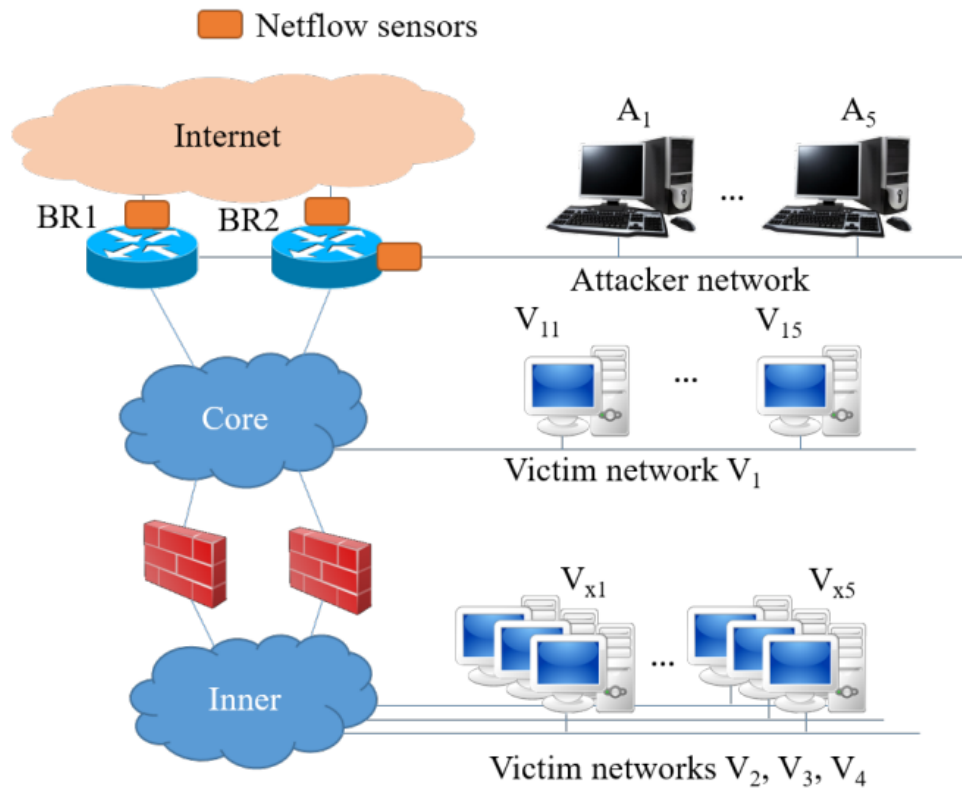


Figure 4.4: Topology of the Network from URG16 [Fernández et al., 2017]

After applying all these operations for the organization of the data, it is now ready to be transformed into a 2D representation. For every second elapsed and for every network, an image that represents the network in that time frame will be created.

### 4.5.1 Feature Selection Approach

Selecting the features according to the nuances of each attack is a critical step in ensuring the efficacy and accuracy of this approach. This process involves a thorough analysis of the characteristics and behaviors associated with the attacks, as well as an examination of the relevant literature and research conducted previously. The selected features for each attack will be detailed in the subsequent sections.

#### Multiple Image Approach

From the research done in chapter 3, we already have an idea of what features might work effectively for each of the attacks. The features indicated in this chapter are just a starting base, as they can change until the end result due to the iterative nature of the DSR methodology employed.

Some features, such as the number of bytes, can possess considerably large values (especially when compared to the values of other features, such as the number of flows). Because of this, all data must be normalized. This ensures that all features have an equal weight in the creation of the histogram values.

### **Port Scanning**

The research performed allowed us to have a deeper knowledge of the best features to select, as can be seen in tables 3.8 and 3.10. This led to the conclusion that for port scanning, the best correlation between feature/attack was with source/destination ports, packet size, and duration of connections [Baldoni et al., 2022].

Based on this, it was decided that the features used in this step are the number of flows, the total size (in bytes), the number of different ports, and the interval between each flow.

### **Denial of Service**

Concerning Denial of Service, the minimal time between two flows, the minimal time between two packets sent, source/destination port, source/destination IP, size of the packets, and "SYN" or "REJ" errors. These would be ideal features, but the dataset does not contain all of these, so total network traffic and its variance could also be considered.

Given the chosen dataset and the available features, the ones that were selected are the size of packets, source/destination IP, minimal time between two flows, and volume of traffic [Baldoni et al., 2022].

### **Singular Image Approach**

Now that the combination of the most effective features for each attack has been determined, an attempt can be made to combine them.

Having in mind that we are upholding a DSR methodology that allows for multiple tests and changes, in this case, to the chosen features; the starting point will be to use both the features selected for the Port Scanning and DoS attacks in their totality. After the first tests regarding the effectiveness of each feature for each attack, a more fundamental conclusion can be reached.

## **4.6 Image Processing Techniques**

After the analysis performed in Chapter 3, it was possible to have an idea about the main characteristics the images created should possess. We should aim to create images that are not too sparse, maintain a sense of spatial awareness, and

are optimized to be fed to a CNN. Until the final method was reached, different schemes were tested, taking full advantage of the DSR approach.

Throughout all the iterations, one aspect remained constant: the flows are grouped in intervals of one second and by subnetwork (42.219.150.\*, 42.219.152.\*, 42.219.154.\*, 42.219.156.\*, 42.219.158.\*).

Something that was noticed throughout this process is that, the dataset does have a huge number of flows because it corresponds to a long period of days, but when grouping the flows by second, in most cases, there were not that many flows, especially when considering that the majority regarded the same few IPs in the network. Even though this can be the case for a smaller-scale network in a real-world scenario, if we want to maintain some sense of spatial awareness in the image, it will always result in a relatively sparse representation.

#### 4.6.1 Aggregated Weighted Feature

This preliminary approach is based on gathering the features selected in subsection 4.5.1 and creating an aggregated and weighted feature value for each IP in the network. The final output is a grayscale image of size 256x256 pixels, where the columns represent the last octet of the destination's IP addresses, and the rows represent the aggregated and weighted feature value.

For the creation of the aggregated feature values, first, the values of each feature must be calculated. We start with an empty array of size 256 for each feature, where each value is connected to a possible destination IP, and iterate through the flows while storing the feature value in the correct cell.

Some of the features selected needed some processing: the size of packets (in bytes) needed to be scaled down to reduce its impact relative to the other features; the minimal time between flows must be inverted, as the relation wanted is that the lower the time is, the more likely an attack.

Once these feature arrays are calculated, they are normalized to the [0-255] range to ensure comparability across different feature magnitudes. It is then possible to combine these arrays, while applying different weights to each feature if beneficial. Now, for each IP in a sub-network it's possible to observe a value that represents the likelihood of an attack.

The weight that each feature can have is something that must be tested and fine-tuned, so the initial weight of all features is equal. Since this Two-Dimensional (2D) representation method was only the starting point of the creation of the method utilized in the approach presented, the search for the ideal weights was not performed; nevertheless, some tests were performed using equal weights.

As it can be seen in the Figures A.1 and A.2, the images created are too sparse, and so, in an attempt to combat this issue, a different method is detailed next.

### 4.6.2 4-Channel Heatmap

The technique proposed for the 2D representation of network flows is based on a heatmap representation. The key principle behind this approach is to encode flow statistics into structured 2D histograms, where the axes represent traffic properties and the pixel intensity encodes the frequency of flows that fall into each bin.

As previously mentioned, all flows are first grouped into 1-second windows and separated by subnet. Within each interval, relevant features were extracted and mapped into four distinct channels. This multi-channel representation provides richer information than a single grayscale aggregation, as each channel emphasizes a different dimension of traffic behavior. The final output for each second is therefore a 4-channel tensor of size  $128 \times 128$ , suitable for input to a CNN.

**Multiple Image Approach for Port Scanning** Because of the inherent characteristics of a port scanning attack, a single 2D heatmap of, for example, destination IP vs destination port, tends to be sparse, especially in a network with modest traffic. Also, collapsing heterogeneous evidence into a single channel can wash out diagnostic cues and retract from the learning ability of a CNN. With this approach we intended to preserve where traffic lands (spatial victim context) and how it behaves through the use of the four different channels.

Following this, the representation possesses three principles:

- **Balanced spatial-behaviour encoding:** maintaining one spatially oriented channel to retain spatial awareness while also adding behaviour-centric channels that remain informative even when the spatial channel is sparse.
- **Axis alignment:** usage of a shared port axis across all channels so that a CNN can learn feature-level filters at the same port coordinates.
- **Densification without hallucination:** usage of sensible binning, percentile clipping, and log transformations to increase the visibility of weaker signals and avoid value inflation.

The first channel encodes the count of flows to each (destination IP, destination Port) cell within the one-second window. The destination IPs are mapped to a coarse row bin of [0-128] deliberately, as using the full /24 grid leads to extremely sparse heatmaps under low traffic, and the spatial identity of the attack is not lost by this action. Scans often present as horizontal strikes (multiple ports on a single destination IP) or vertical strikes (same destination port across multiple destination IPs), as shown in Figure A.8.

Channel two surfaces protocol-level intent. Each flow's TCP flags are analyzed, checking to see which flags are present. Port scanning attacks are dominated by flows with only SYN flags, so the count of the flows that have this characteristic is placed on the vertical axis, while the horizontal axis remains the destination port, as by the representation principle stated above.

Channel three entails the temporal behavior of the network. Port scanning attacks often generate burst, regular sequences of traffic, and so the number of flows that fall between the normalized time interval between flows is presented on the vertical axis.

Lastly, channel four represents the payload footprint of the network. Since these attacks tend to have small and consistent packets, the number of flows with the same size is calculated. This calculated value is spread through the vertical axis while being normalized through the average size of the packets present in the window.

After the creation of each channel, they are normalized independently, and percentile clipping is applied. Global scaling was not applied because, as previously mentioned, the dataset has long periods of very sparse flows; this would mostly lead to a sparser image. With the channels, a tensor of shape (4, 128, 128) is created, and the resulting images have a size of 64KB.

Figures A.7 and A.8 present the four heatmaps created for a benign and port scan, respectively.

**Multiple Image Approach for Denial of Service** Since this is the same approach as the one detailed above, but directed to the detection of DoS attacks, the same structure and key principles are applied. The shared horizontal axis across all channels is now the destination IPs instead of the destination ports, as this information is much more relevant for this application. The main differences appear in the construction of the channels.

For the first channel the source IPs were binned on the vertical axis. This approach is focused on DoS attacks, meaning that a single machine usually performs it. This means that only a single cell that represents the attack on a victim's column is going to be filled (other cells in the column can be filled, but they would be a by-product of benign flows); nevertheless this can be a good indicator, as the cell will present a very high intensity based on the high number of flows that are mapped there in a DoS instance (this can be analyzed in Figure A.10 at around the eighth column). This channel representation means that, in the event that we also wanted to detect a Distributed Denial of Service (DDoS) attack, a victim's column would be filled with a high number of cells, each of high intensity, producing a unique signature.

Channel two is similar to the one presented in the Port Scanning attack approach detailed above. It represents protocol-level behavior in the network by analyzing the flags present in each flow. Flows that possess either a SYN flag or an RST flag and no corresponding ACK flag are valued.

The third channel is equal to the one detailed above for port scanning. It entails analyzing the temporal behavior of the network by examining the time interval between each flow within a one-second window. This is advantageous, as a typical DoS attack produces a high volume of flows, and this representation leads to a very noticeable attack signature, as shown in Figure A.10.

The fourth channel continues to have the destination IPs on the horizontal axis, and the vertical axis represents the logarithmically binned average packet size. Each cell holds the number of flows whose average packet size fell into that size row and destination IP. Since a typical DoS is a burst of repetitive packets, only a row (or a few adjacent ones) will be filled for a victim's destination IP. This is a good attack indicator, as benign data most usually contains various packet sizes between flows, leading to a more crowded vertical axis. This can be observed in Figures A.9 and A.10.

A logarithmic function is applied to channels one and four to contradict their heavy-tailed nature lightly. Percentile clipping is applied to non-zero cells alongside normalization. The tensor of shape (4, 128, 128) can now finally be created.

**Single Image Approach** Since an image for the detection of each attack individually has already been created, the challenge now lies in how to combine these two into one. The four channels created must retain enough information for the detection of either one of the attacks. Channels two and three of both the previous approaches are already really similar, as they both contain relevant information for the detection of each attack. Regarding channels one and four, a middle ground that is satisfactory for both attacks is proposed. For the retention of spatial awareness through the channels, the shared horizontal axis represents the destination IPs. Utilizing destination ports as the shared axis would be too restrictive for the detection of DoS attacks. The same count of flows for the value of each cell in the channels is utilized.

The first channel is focused more on the detection of port scans, as the rows are mapped to the destination ports. This allows for port scans to have a very noticeable signature, as for a victim's destination IP, a vertical stripe will appear representing all the different ports scanned, as shown in Figure A.5. In a DoS attack instance, most likely only a cell on the victim column will be filled with high intensity, as these attacks tend to focus on a specific port.

Channels two and three remain as representations of protocol-level and temporal behavior of the network, with the same construction as previously detailed. Channel four is more oriented for the detection of DoS attacks, and as the same format as the one created with the intent of only detecting this attack. Nevertheless, it is also relevant for the detection of port scans, as these tend to be comprised of small and consistent packets.

Looking at Figure A.3, which represents only benign traffic, it's possible to detect that a particular IP was actually receiving a lot of traffic. In this case, the other three channels allow us to immediately understand that no attack is present, as channel one does not present the vertical signature of a port scan attack, channel 2 shows that the majority of flows contains the ACK flag, and channel 4 shows a spread of flows with different packet sizes in the same destination IP; this indicates that no attack is present in this window.

Figures A.3, A.5, and A.4 present the four heatmaps created for a benign, port scan, and DoS, respectively.

## 4.7 Deep Learning Model

CNNs are widely used in image processing due to their ability to automatically and adaptively learn spatial hierarchies of features, as described in more detail in Section 3.4.1. The model's architecture, training procedures, and evaluation metrics are going to be presented next. To evaluate our system, we will use metrics such as accuracy, precision, recall, and AUC [Taheri et al., 2018].

The architecture includes convolutional and pooling layers for feature extraction, followed by fully connected layers for classification. It is trained using the Adam optimizer, which is an adaptive learning rate optimization algorithm that is well-suited for deep learning applications. The model was trained for 10 epochs, and early stopping was used. In each epoch, the model was fitted on the training data and evaluated on the test data. The number of steps per epoch was determined by the length of the data generators, ensuring the model processes the entire dataset in each epoch.

### 4.7.1 Addressing Class Imbalance

As it was discussed previously, the UGR'16 dataset provides multiple CSV files of the networks data that have already been preprocessed and labeled. It also provides files that contain only the attack flows, organized by timestamps. In the first iteration of this approach the images that represent the network under attack were created through the use of these attack only flow files, allowing us to create the maximum number of images that represented the attack since we already knew that creating the images for the entirety of the dataset would lead to a major imbalance in the number of images per class (more so between benign and attack).

The problem with this approach was that it was misrepresenting the state of the network when creating these attack labeled images by completely disregarding the flows that represented everything else that was happening in the network in that particular time frame. Having in mind the purpose of this work, creating these types of images will not be sufficient, although it might have led to some better results in this context, where we have all the information pre-labeled and preprocessed, the trained models would be of little use in a real-world scenario.

To combat this issue, the timestamps of the beginning of the attacks were gathered through the use of these attack-only CSV files and utilized to only process images around those time frames through the file that contained the entirety of the flows. This way, we managed to keep a balance of the images created for each class and also make sure that the integrity of the network is not lost in the creation of any image.

Some representative samples of benign images were also drawn from non-attack intervals across the different weeks in an attempt to maintain some temporal diversity.

## 4.8 Summary

This chapter outlines the proposed methodology for the creation of a NADS that focuses on the detection of DoS and Port Scanning attacks. The methodology's structure is based on a DSR approach, emphasizing real-world problem-solving, artifact creation, iterative processes, and communication of design knowledge.

The organization and timeline of the work conducted until the Intermediate Delivery and the Final Delivery is presented in GANTT charts.

The URG'16 dataset is chosen for its suitability, meeting criteria such as the type of attack present, sufficient data volume, and satisfactory attack time window.

Features that are going to be used for each attack were chosen, the data cleanup process was presented, and the transformation to a 2D representation was also detailed. It's important to note that, based on the DSR interactive process approach, a lot of details may change throughout the practical implementation stage and so, none of these decisions are considered final.

This chapter sets the stage for the practical implementation that is going to be performed in the following months, highlighting the approach, key decisions, and considerations for each stage of the development.

Multiple adjustments had to be performed to correctly handle the dataset data. According to the official documentation of the dataset, it captured the data from multiple networks and so, in order for our solution to work, the traffic must be separated. Because of this, the first operation performed on the dataset was to group the traffic of all four sub-networks. All of the remaining operations are performed for each one of these groups individually.

# Chapter 5

## Experiments and Results

In this chapter, the iterative steps will be described, starting from the base established decisions stated in chapter 4 until the final results. Numerous results will be shown, and it is crucial to note that the values were derived by averaging multiple runs (a minimum of three runs were performed for each scenario). This approach was taken to reduce the impact of any possible outliers created from individual runs.

Following the methodology detailed in 4-Channel Heatmap 4.6.2, the images created were fed into the Convolutional Neural Network (CNN). They are split 80-20 into a training and a testing dataset. The class distribution of the created images is presented in Figures 5.1, 5.2, and 5.3. Ideally, the number of images in each class would be considerably higher, but we are limited by the number of attack flows present in the dataset. Even when applying the principles detailed in subsection 4.7.1 that allow us to gather all the attack flows from the entire dataset without computing an insane amount of benign images, the number of images, especially for port scanning attacks, is small.

Even though a much larger number of benign images could have been used, it would only enlarge the class imbalance, and so only around double the amount of benign images were used in relation to the amount of attack images.

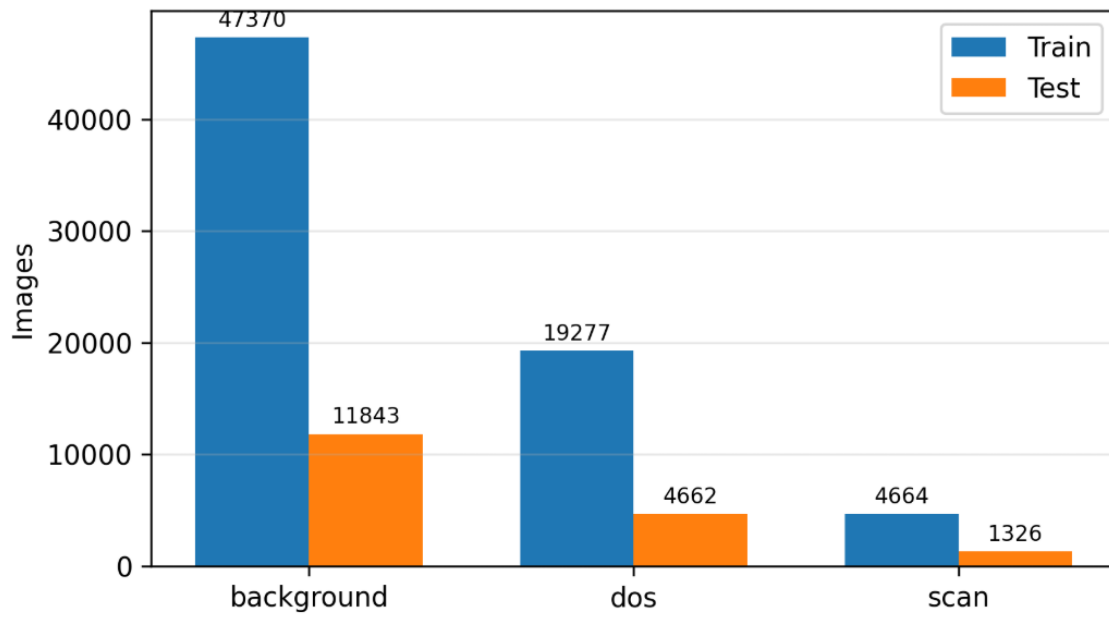


Figure 5.1: Class Distribution for Single Image Approach

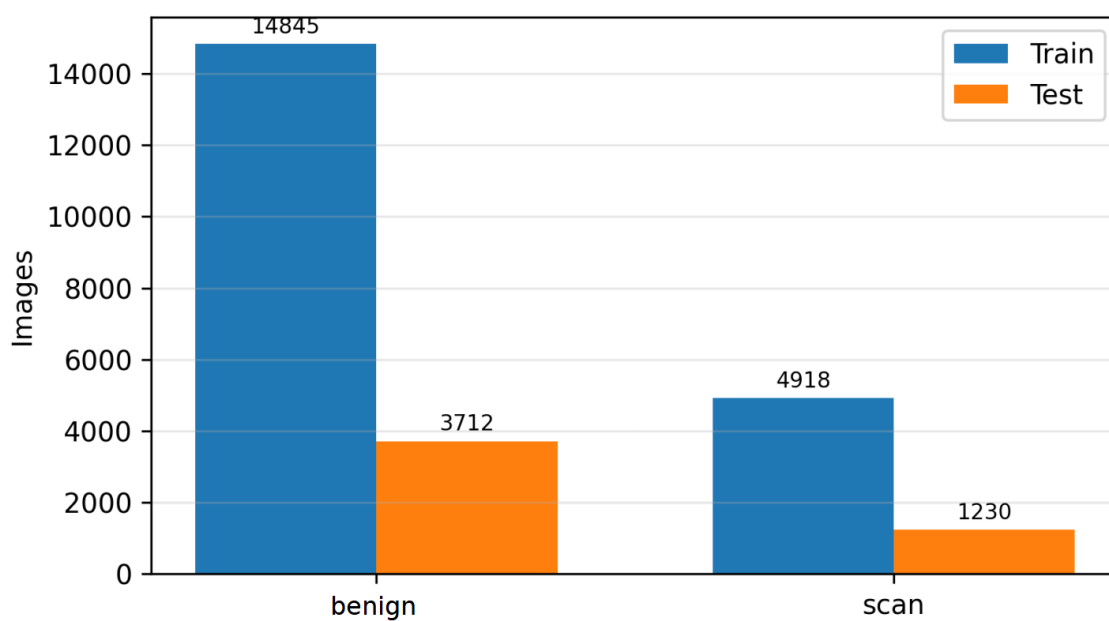


Figure 5.2: Class Distribution for Port Scanning

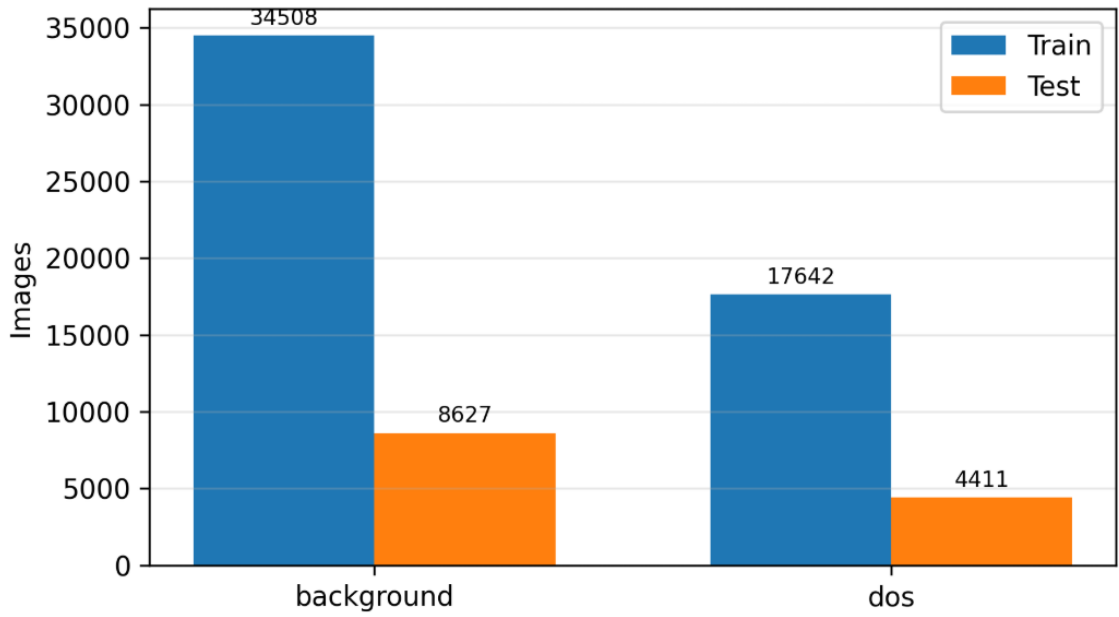


Figure 5.3: Class Distribution for Denial of Service

## 5.1 Single Image Approach

The results obtained from this approach are concerningly positive, as shown in Figures 5.4 and 5.5. The model shows that it can correctly identify every attack instance, without creating any False Positives (FPs) or False Negatives (FNs). This behavior raises some concern about the legitimacy of the results obtained.

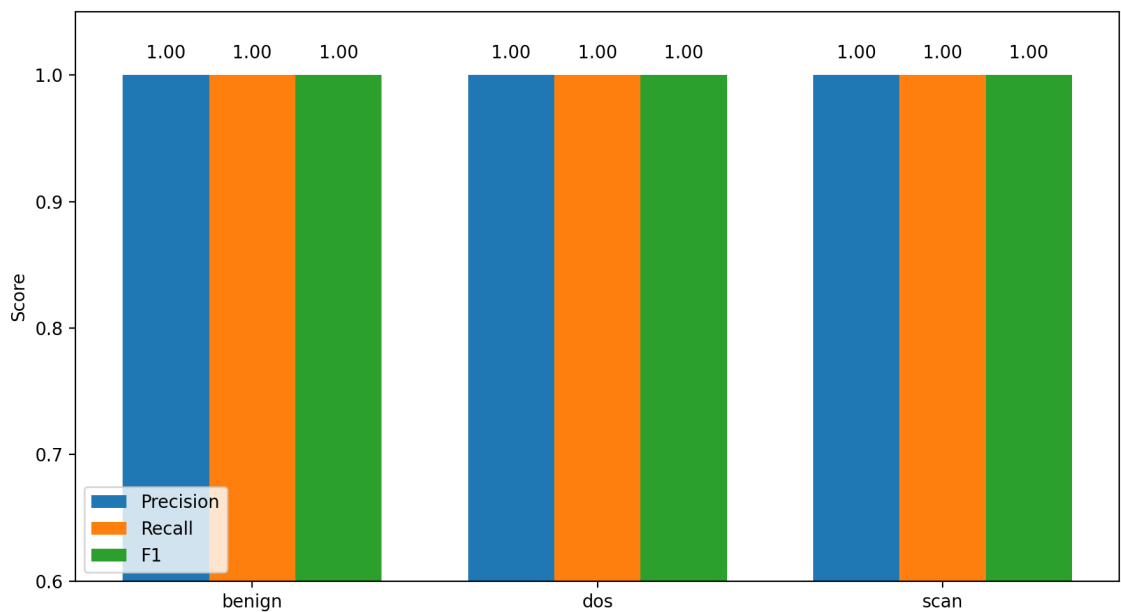


Figure 5.4: Precision, Recall, and F1 Metrics for Single Image Approach

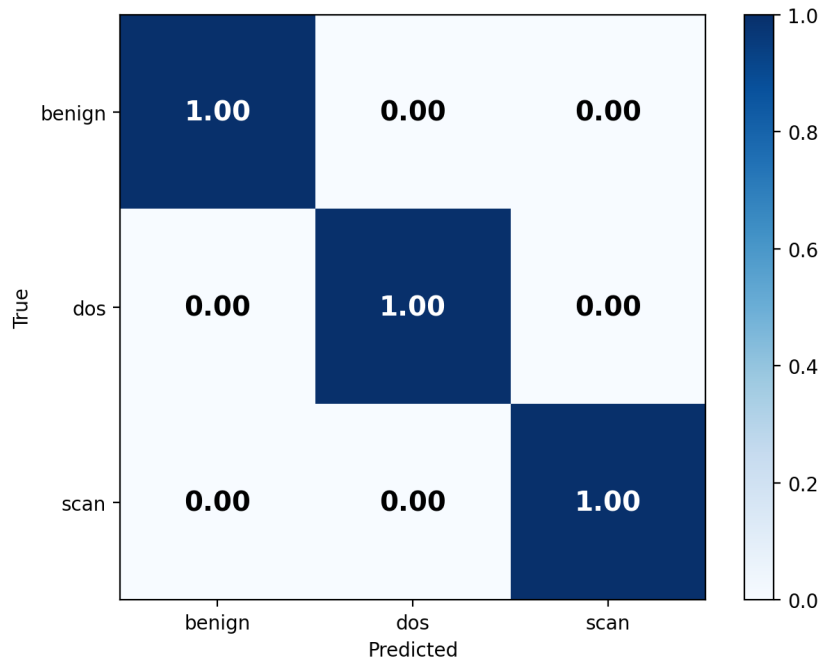


Figure 5.5: Confusion Matrix for Single Image Approach

To confirm that the model is working correctly and that the resulting metrics aren't a result of an error within the CNN, we performed some verifications. A few images that were created and correctly labeled as benign were added to both the attack image pools in the testing dataset. As we can see from Figures 5.6 and 5.7, the added images create false positives and lower the values of all the metrics. This behavior from the model is precisely what we expected: when presented with a benign image mixed with the attack ones, the model managed to correctly label them as being benign, triggering false positives.

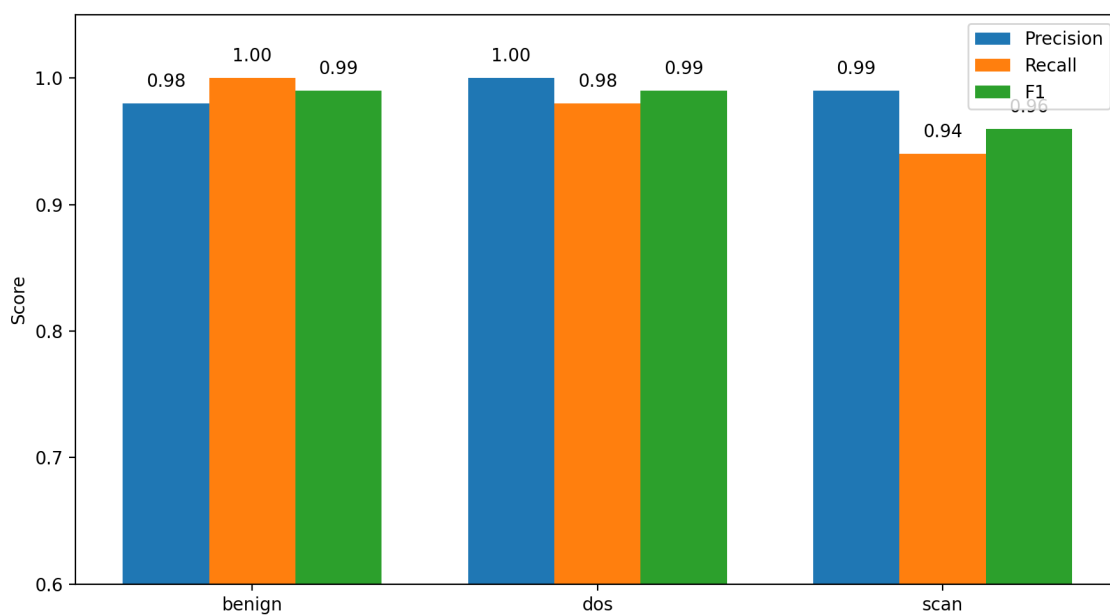


Figure 5.6: Skewed Metrics

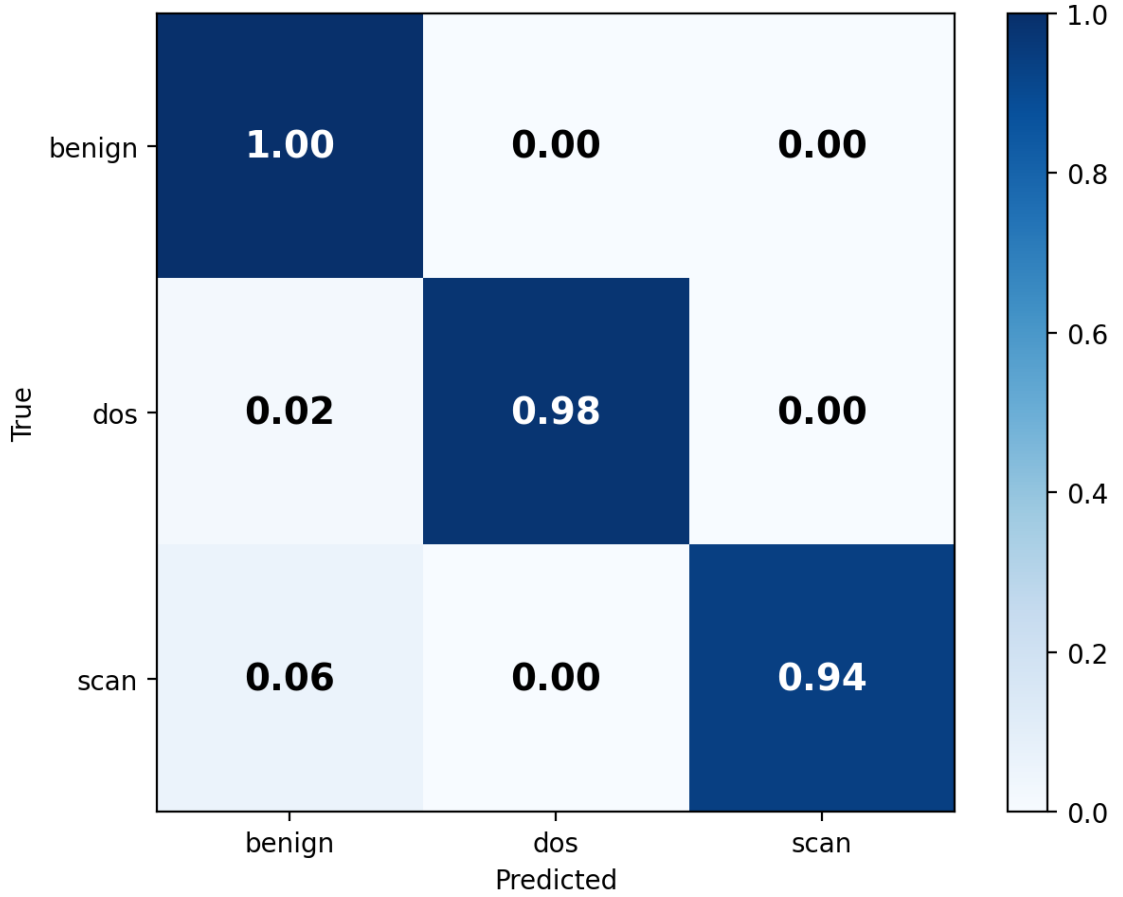


Figure 5.7: Skewed Confusion Matrix

As mentioned at the beginning of this chapter, it was only possible to create a small number of attack images. This means that the model isn't exposed to sufficient attack images, leading to possible generalization difficulties when applied in a real-world scenario. This can also be a motive for why the model is performing so well.

The structure of each attack instance is also maintained, meaning that even though there are multiple instances of each attack spread over multiple weeks, their Two-Dimensional (2D) representation is reasonably similar. Every Denial of Service (DoS) attack has a duration of three minutes, a packet size of 1280 bits, and a 100 packets/s rate. Every port scan instance also has a duration of three minutes, and the same victims are affected.

These two factors make it challenging for the model to achieve a good level of generalization, which can explain why the presented metrics are so high. When the attacks possess the structure that the model is designed to detect, it can perfectly label all its instances. However, even though we can't test this using the URG'16 dataset, this leads us to believe that its generalization suffers.

Nevertheless, as we can see in the Figures A.4 and A.5, the 2D representation does create an apparent attack signature for each attack; even with the generalization problem, it is clear that the different channels are providing significant context

about the attacks.

Figure A.6 shows the per-class average of the 4-channel images created. Since these are means over many images, the majority of the channels are filled with empty cells. The only thing that's noticeable is the attack signature for both attacks. The fact that in every channel, the attack signature is only present for a single destination IP explains the size of the generalization problem.

## 5.2 Multiple Image Approach

### 5.2.1 Port Scanning

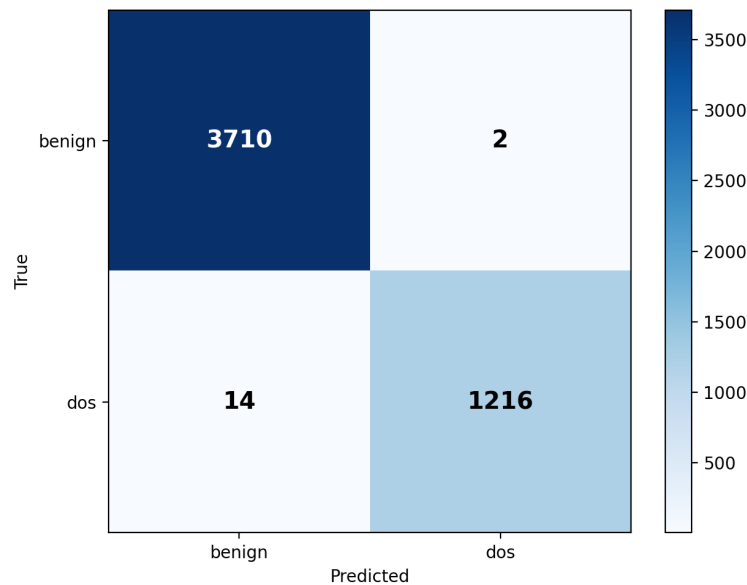


Figure 5.8: Confusion Matrix for Port Scanning

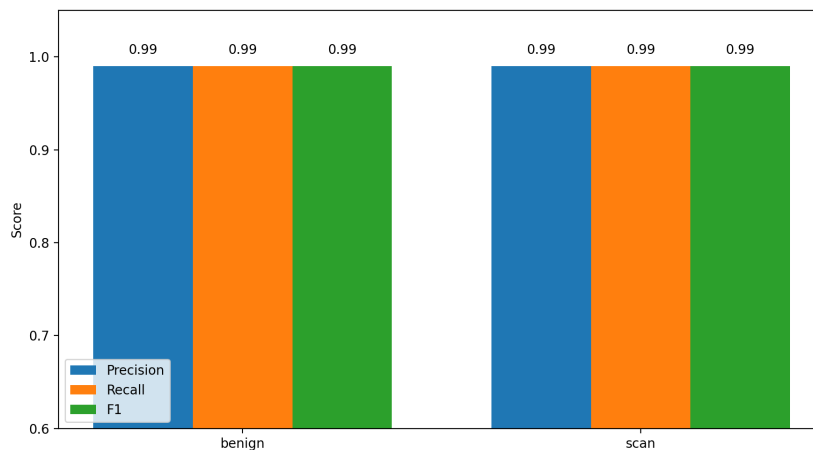


Figure 5.9: Precision, Recall, and F1 Metrics for Port Scanning

The fact that this approach performs worse than the Single Image approach leads us to conclude that a specific channel in this approach is creating confusion between the benign and attack images. We can see from Figure 5.8 that the model creates 14 false positives and two false negatives. In the event that a machine has multiple services running on different ports, the model struggles to differentiate it from an instance of a port scan where only a few ports were probed. The attack signature present in the channels can be seen in Figure A.11.

The port scans performed were all fast-paced attacks, meaning that in the one-second window utilized, enough ports were scanned for an attack signature to be present in the images created. For a slower-paced attack, the model will have a lot of difficulties in its detection, as the 2D representation will be very similar between attack and benign instances.

### 5.2.2 Denial Of Service

In this approach, the signatures of attack present in each channel are mostly of a single cell having an extremely high intensity. This makes it a lot harder for the CNN to learn visual features from the channels, especially when compared to the horizontal/vertical stripes that port scans have. It was expected that this approach would result in the worst metrics compared to the others, and, as shown in Figure 5.10, it is the one that created more FP and FN.

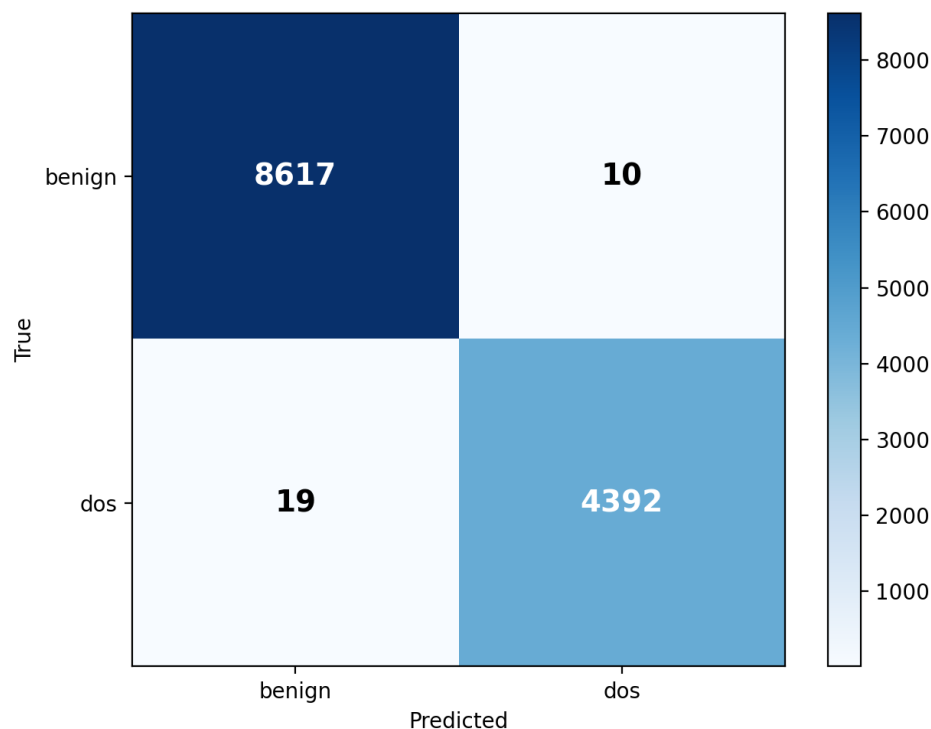


Figure 5.10: Confusion Matrix for Denial of Service

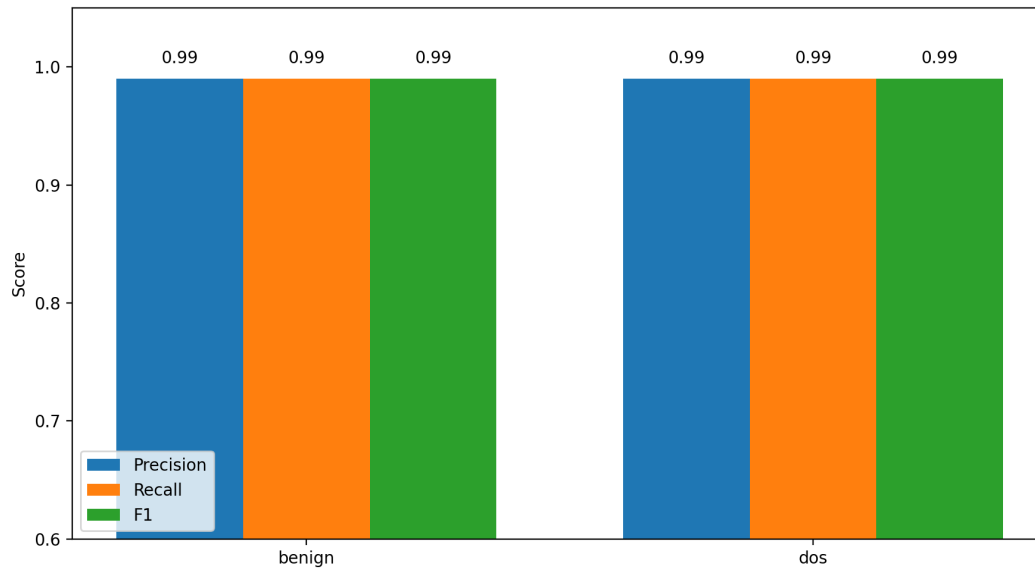


Figure 5.11: Precision, Recall, and F1 Metrics for Denial of Service

The attack signature present in the channels of each image for channels 1 and 4 is an isolated cell with extremely high intensity. We can see this in the images that were created and that are presented in Figures A.9, A.10, and A.12. In these channels, the benign flows are represented as multiple low-intensity cells on the same column (destination IP). For instances where a low flow count is present in the one-second window, the benign and attack representations in these channels can become extremely similar, leading to false positives. In the event that a machine in the subnetwork only receives flows from a single source, all with similar packet sizes, the only thing that differentiates this from being a representation of an attack instance in channels 1 and 4 is the cell intensity. Putting all this together, we can understand why the model is presenting some false positives.

Based on the nature of the attack, instances where a machine is simply being highly requested can be mislabeled, leading to false negatives.

Even with all of these setbacks, the metrics observed in Figure 5.11 are highly positive. Once again, this is most likely due to the generalization problem described above.

### 5.3 Summary

The results highlight a trade-off between generality and specialization. The Single Image Approach achieves perfect metrics within UGR'16, correctly classifying all instances of DoS and Port Scanning. However, this raises concerns of overfitting and limited generalization, since the dataset's attack instances are repetitive in duration, packet size, and victim set.

It was expected that the results for the multiple image approach would be better than for the single image approach, since it allows the usage of the most descrip-

| Approach       | Attack Type   | Accuracy | Precision | Recall | F1 Score |
|----------------|---------------|----------|-----------|--------|----------|
| Single Image   | DoS + Scan    | 1.0      | 1.0       | 1.0    | 1.0      |
| Multiple Image | Port Scanning | 0.99     | 0.99      | 0.99   | 0.99     |
| Multiple Image | DoS           | 0.99     | 0.99      | 0.99   | 0.99     |

Table 5.1: Comparative Performance of the Different Approaches

tive feature channel for each attack. This was not the case as shown in Table 5.1. A better description of the specific attack instance can be provided with this approach, but it also allows for edge cases where the benign and attack representations become too similar.

Overall, the comparative results suggest that a single unified model can learn dataset-specific attack structures very effectively. This aligns with the broader conclusion that balanced multi-channel heatmaps are effective, but their robustness must be validated on more diverse datasets.



# Chapter 6

## Conclusion

This thesis set out to explore an approach to network anomaly detection by leveraging image processing techniques to transform network traffic flows into balanced multi-channel heatmaps, subsequently analyzed through Convolutional Neural Networks (CNNs). Motivated by the limitations of traditional rule-based and signature-based systems, the work demonstrated that Two-Dimensional (2D) representations of flows can serve as expressive abstractions of network behavior, enabling machine learning models to effectively distinguish between benign and malicious traffic.

By following a Design Science Research (DSR) methodology, the study iteratively refined the representation strategy, beginning with single-feature aggregations and culminating in a balanced four-channel heatmap design tailored for Denial of Service (DoS) and Port Scanning detection. The experiments conducted with the UGR'16 dataset confirmed that the proposed representations successfully capture distinctive attack signatures, allowing even relatively simple CNN architectures to achieve near-perfect classification metrics. Importantly, the research highlights a trade-off between specialized per-attack representations and unified representations.

While the results obtained in this work were consistently strong, it is crucial to acknowledge the limitations regarding generalization. The UGR'16 dataset, despite its scale and realism, presents attack instances that are highly structured and repetitive in terms of duration, packet size, and victim distribution. As a result, the models trained in this context may have partially learned to recognize these recurring patterns rather than the broader behavioral traits of DoS and Port Scanning attacks. This raises concerns about the system's ability to generalize when exposed to attacks with different dynamics. Therefore, although the performance metrics demonstrate the feasibility of the approach, they should be interpreted cautiously, with an understanding that real-world deployments require extensive validation across diverse datasets and conditions.

## 6.1 Future Work

While the proposed system achieved promising results in detecting DoS and Port Scanning attacks, several avenues remain open for future exploration and improvement. While analyzing the metrics of the CNNs created, we concluded that the model would struggle to generalize properly. By performing a cross-dataset evaluation, testing the model against other datasets that include both the attacks, its generalization can be analyzed to confirm whether the model is currently overfit to the attack structure present in the UGR'16 dataset. Conversely, training on multiple datasets simultaneously could highlight which elements of the image representation are general and which are specific to a single dataset.

Another promising avenue for future exploration is the adoption of hybrid architectures that combine CNNs with Long short-term memory (LSTM). CNNs are highly effective at extracting spatial features from the 2D heatmap representations of network traffic, capturing localized attack signatures such as horizontal stripes in port scans. However, they are inherently limited in their ability to model temporal dependencies across consecutive time windows. LSTMs, on the other hand, are designed to learn long-term sequential patterns, making them well-suited to capture the temporal evolution of network behavior. By integrating CNN layers for spatial feature extraction with LSTM layers for temporal sequence modeling, a hybrid CNN–LSTM model could provide a richer representation of the attacks. Such a model may outperform purely spatial or purely sequential approaches, particularly in detecting slow-evolving scans or multi-stage attacks that unfold over extended periods.

# References

- Iftikhar Ahmad, Azween B Abdullah, and Abdullah S Alghamdi. Application of artificial neural network in detection of probing attacks. In *2009 IEEE Symposium on Industrial Electronics & Applications*, volume 2, pages 557–562, 2009. doi: 10.1109/ISIEA.2009.5356382.
- Esra Altulaihan, Mohammed Amin Almaiah, and Ahmed Aljughaiman. Anomaly detection ids for detecting dos attacks in iot networks based on machine learning algorithms. *Sensors*, 24(2), 2024. ISSN 1424-8220. doi: 10.3390/s24020713. URL <https://www.mdpi.com/1424-8220/24/2/713>.
- Sara Baldoni, Marco Carli, and Federica Battisti. Analysis of a 2d representation for cps anomaly detection in a context-based security framework. *Frontiers in Signal Processing*, 1, 1 2022. doi: 10.3389/frsip.2021.814129.
- Gueltoum Bendiab, Stavros Shiaeles, Abdulrahman Alruban, and Nicholas Kolokotronis. Iot malware network traffic classification using visual representation and deep learning. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pages 444–449, 2020. doi: 10.1109/NetSoft48620.2020.9165381.
- Bo Cao, Chenghai Li, Yafei Song, Yueyi Qin, and Chen Chen. Network intrusion detection model based on cnn and gru. *Applied Sciences*, 12(9), 2022. ISSN 2076-3417. doi: 10.3390/app12094184. URL <https://www.mdpi.com/2076-3417/12/9/4184>.
- Sofia Casarin, Sara Baldoni, Marco Carli, Pietro Zanuttigh, and Federica Battisti. Unsupervised network anomaly detection by learning on 2d data representations. In *2022 9th Swiss Conference on Data Science (SDS)*, pages 53–58, 2022. doi: 10.1109/SDS54800.2022.00016.
- Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey, 2019. URL <https://arxiv.org/abs/1901.03407>.
- Ravi Chauhan and Shahram Shah Heydari. Polymorphic adversarial ddos attack on ids using gan. In *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–6, 2020. doi: 10.1109/ISNCC49221.2020.9297264.
- Aldo Cortesi. binvis. <https://binvis.io/#/>, 2015.
- Gideon Creech and Jiankun Hu. Generation of a new ids test dataset: Time to retire the kdd collection. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 4487–4492, 2013. doi: 10.1109/WCNC.2013.6555301.

- T Cruz, J Proença, P Simões, M Aubigny, M Oedraogo, A Graziano, and L Yasakhetu. Improving cyber-security awareness on industrial control systems: The CockpitCI approach. *Journal of Information Warfare*, 13(4):27–41, 2014. ISSN 14453312, 14453347.
- Khaled Elleithy, Drazen Blagovic, Wang Cheng, and Paul Sideleau. Denial of service attack techniques: Analysis, implementation and comparison. *Journal of Systemics, Cybernetics and Informatics*, 3:66–71, 01 2006.
- Gabriel Maciá Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Theron. Ugr’16: A new dataset for the evaluation of cyclostationarity-based network idss, 11 2017.
- Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of the 6th International Conference, Co-NEXT ’10*, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450304481. doi: 10.1145/1921168.1921179. URL <https://doi.org/10.1145/1921168.1921179>.
- Sebastián García, Martin Grill, Jan Stiborek, and Alejandro Zunino. An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123, 09 2014. doi: 10.1016/j.cose.2014.05.011.
- Oleg Gorokhov, Mikhail Petrovskiy, and Igor Mashechkin. Convolutional neural networks for unsupervised anomaly detection in text data. In Hujun Yin, Yang Gao, Songcan Chen, Yimin Wen, Guoyong Cai, Tianlong Gu, Junping Du, Antonio J. Tallón-Ballesteros, and Minling Zhang, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2017*, pages 500–507, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68935-7.
- Sakib Haque, Zachary Eberhart, Aakash Bansal, and Collin McMillan. Semantic similarity metrics for evaluating source code summarization. In *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, ICPC ’22*, page 36–47, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392983. doi: 10.1145/3524610.3527909. URL <https://doi.org/10.1145/3524610.3527909>.
- Bruce Hartpence and Andres Kwasinski. Combating tcp port scan attacks using sequential neural networks. In *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 256–260, 2020. doi: 10.1109/ICNC47757.2020.9049730.
- Mahdi Nsaif Jasim, Ali Munther Abdul Rahman, and Muthanna Jabbar Abdulredhi. Machine learning classification-based portscan attacks detection using decision table. *Indonesian Journal of Electrical Engineering and Computer Science*, 29(3):1466–1472, March 2023. ISSN 2502-4752. doi: 10.11591/ijeecs.v29.i3.pp1466-1472. URL <http://ijeecs.iaescore.com>.
- P Jeya, Ravichandran Muralidharan, and Ravichandran C S. Efficient classifier for r2l and u2r attacks. *Int. J. Comput. Appl.*, 45, 01 2012.

- Atul Kant Kaushik, Emmanuel S. Pilli, and R.C. Joshi. Network forensic system for port scanning attack. In *2010 IEEE 2nd International Advance Computing Conference (IACC)*, pages 310–315, 2010. doi: 10.1109/IADCC.2010.5422935.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25:1097–1105, 01 2012. doi: 10.1145/3065386.
- Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C. Suh, Ikkyun Kim, and Kuinam J. Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22:949–961, 1 2019. ISSN 15737543. doi: 10.1007/s10586-017-1117-8.
- MIT Lincoln Lab. 1998 DARPA Intrusion Detection Evaluation Dataset. <https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset>, 1998a.
- MIT Lincoln Lab. 1999 DARPA Intrusion Detection Evaluation Dataset. <https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset>, 1998b.
- Andrew Moore, Denis Zuev, and Michael Crogan. Discriminators for use in flow-based classification, rr-05-13. Technical report, Queen Mary University of London, 08 2005.
- Andrew W Moore and Denis Zuev. Internet traffic classification using bayesian analysis techniques, 6 2005.
- Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, 2015. doi: 10.1109/MilCIS.2015.7348942.
- Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review, 4 2021. ISSN 15577341.
- Iñigo Perona, Ibai Gurrutxaga, Olatz Arbelaitz, José I. Martín, Javier Muguerza, and Jesús Ma Pérez. Service-independent payload analysis to improve intrusion detection in network traffic. In *Proceedings of the 7th Australasian Data Mining Conference - Volume 87, AusDM '08*, page 171–178, AUS, 2008. Australian Computer Society, Inc. ISBN 9781920682682.
- Stolfo S. KDDCup'99. <https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 1998.
- Ali Shiravi, Hadi Shiravi, Mahbod Tavallaei, and Ali Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & Security*, 31:357–374, 05 2012. doi: 10.1016/j.cose.2011.12.012.
- Seong So and A L Narasimha Reddy. A study of analyzing network traffic as images in real-time, 4 2005.

- 
- Anna Sperotto, Ramin Sadre, Frank van Vliet, and Aiko Pras. A labeled data set for flow-based intrusion detection. In Giorgio Nunzi, Caterina Scoglio, and Xing Li, editors, *IP Operations and Management*, pages 39–50, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04968-2.
- Shayan Taheri, Milad Salem, and Jiann Shiun Yuan. Leveraging image representation of network traffic data and transfer learning in botnet detection. *Big Data and Cognitive Computing*, 2:1–16, 12 2018. ISSN 25042289. doi: 10.3390/bdcc2040037.
- Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6, 2009. doi: 10.1109/CISDA.2009.5356528.
- Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng. Malware traffic classification using convolutional neural network for representation learning. In *2017 International Conference on Information Networking (ICOIN)*, pages 712–717, 2017. doi: 10.1109/ICOIN.2017.7899588.
- Meng Yue, Huaiyuan Wang, Liang Liu, and Zhijun Wu. Detecting dos attacks based on multi-features in sdn. *IEEE Access*, 8:104688–104700, 2020. doi: 10.1109/ACCESS.2020.2999668.
- Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V. Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data, 2018. URL <https://arxiv.org/abs/1811.08055>.

# Appendices





# Appendix A

## 2D Representations

### A.1 Aggregated Weighted Feature



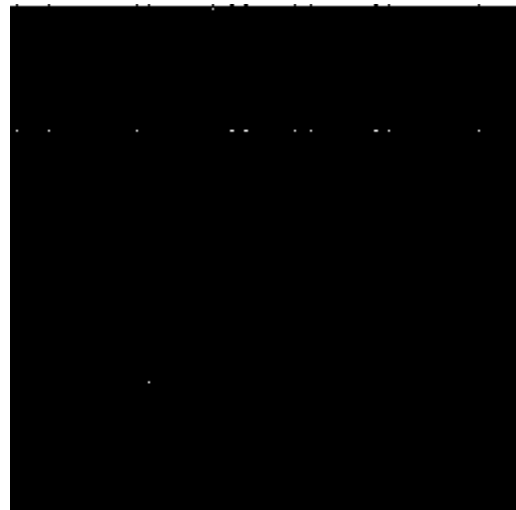
Multiple Image Approach for DoS Benign Image



Multiple Image Approach for DoS Attack Image



Multiple Image Approach for Port Scanning Benign Image



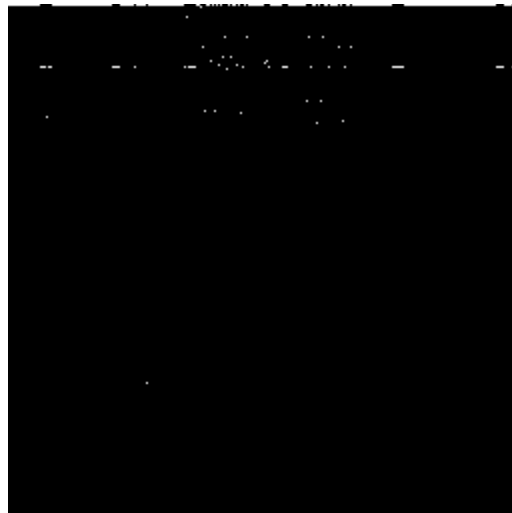
Multiple Image Approach for Port Scanning Attack Image



Single Image Approach Benign Image



Single Image Approach DoS Attack Image



Single Image Approach Port Scanning Attack Image

Figure A.2: Comparison of Single Image Approach Images.

## A.2 4-Channel Heatmap

### A.2.1 Single Image Approach



Figure A.3: Benign for Single Image Approach



Figure A.4: Denial of Service for Single Image Approach



Figure A.5: Port Scan for Single Image Approach

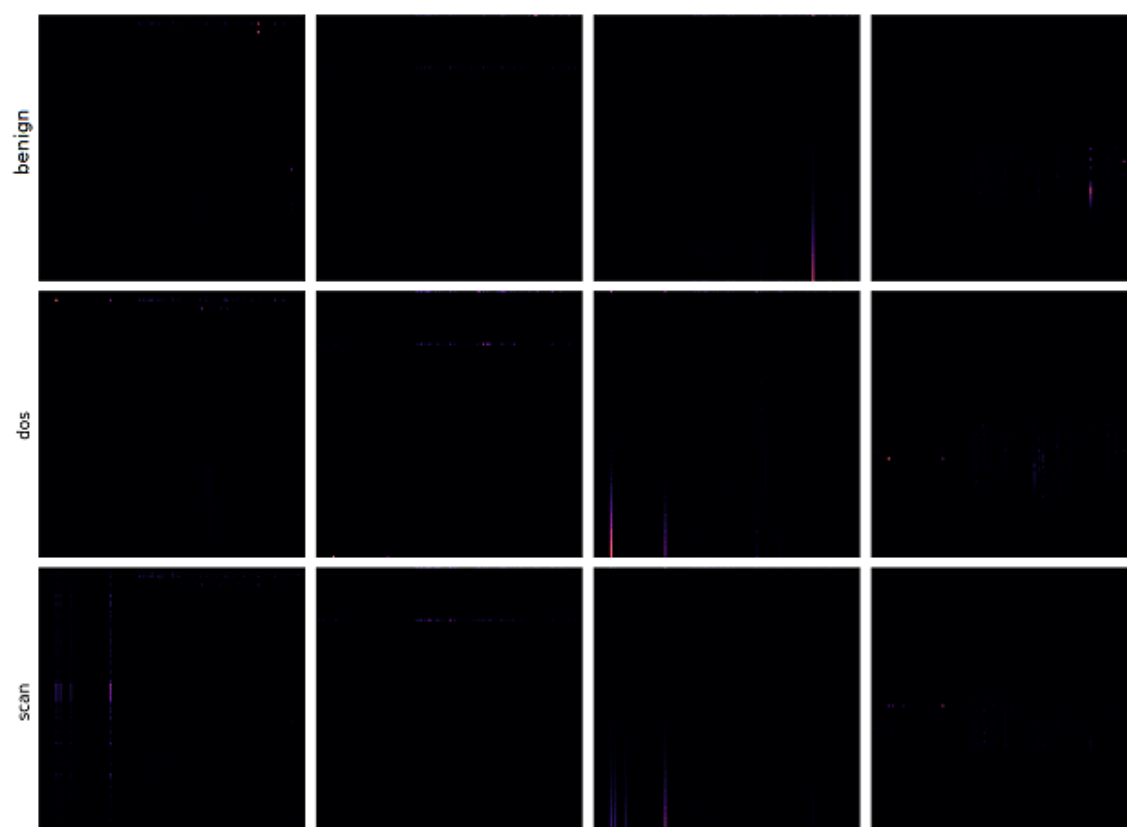


Figure A.6: Mean Channel Images per Class for Single Image Approach

## A.2.2 Multiple Image Approach



Figure A.7: Benign for Port Scanning

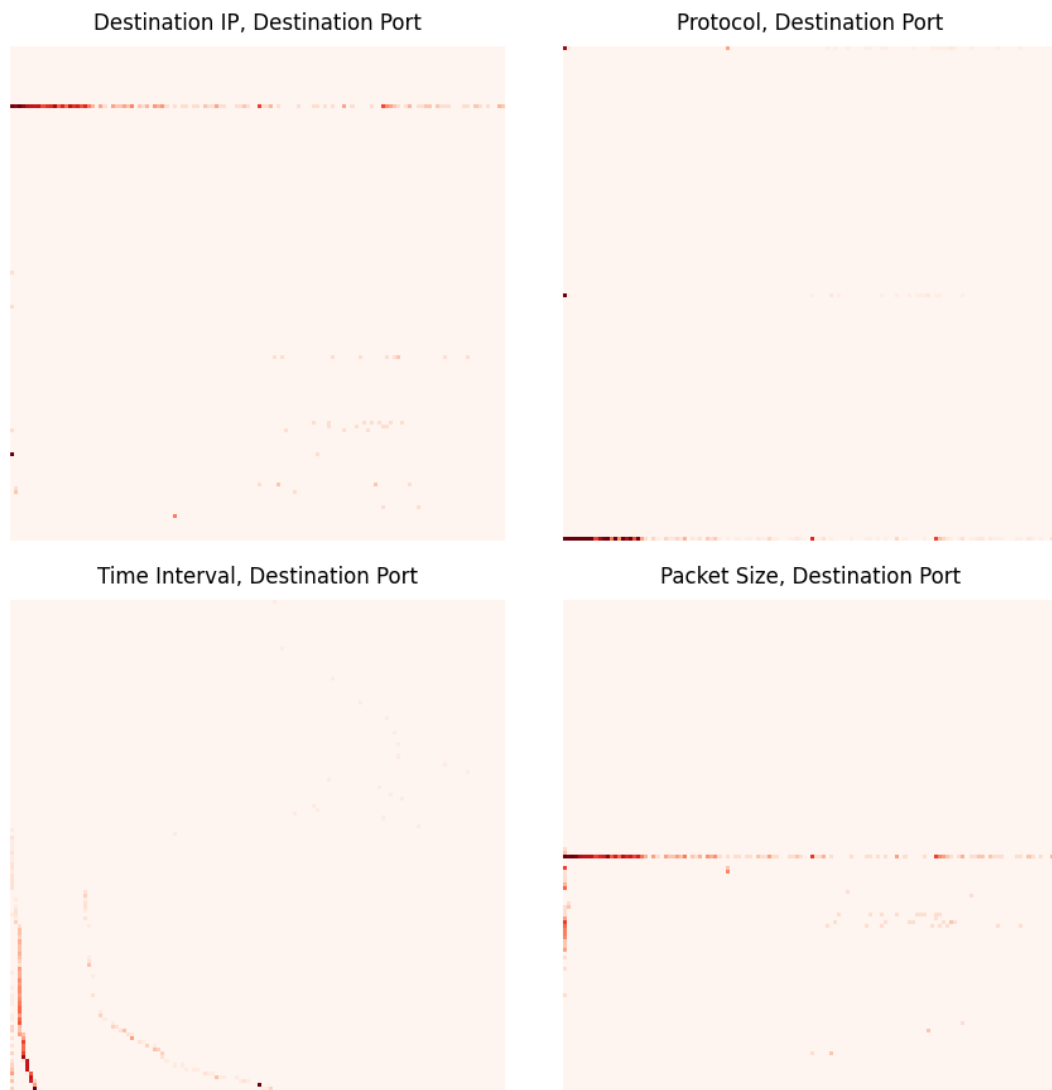


Figure A.8: Attack for Port Scanning



Figure A.9: Benign for Denial of Service



Figure A.10: Attack for Denial of Service

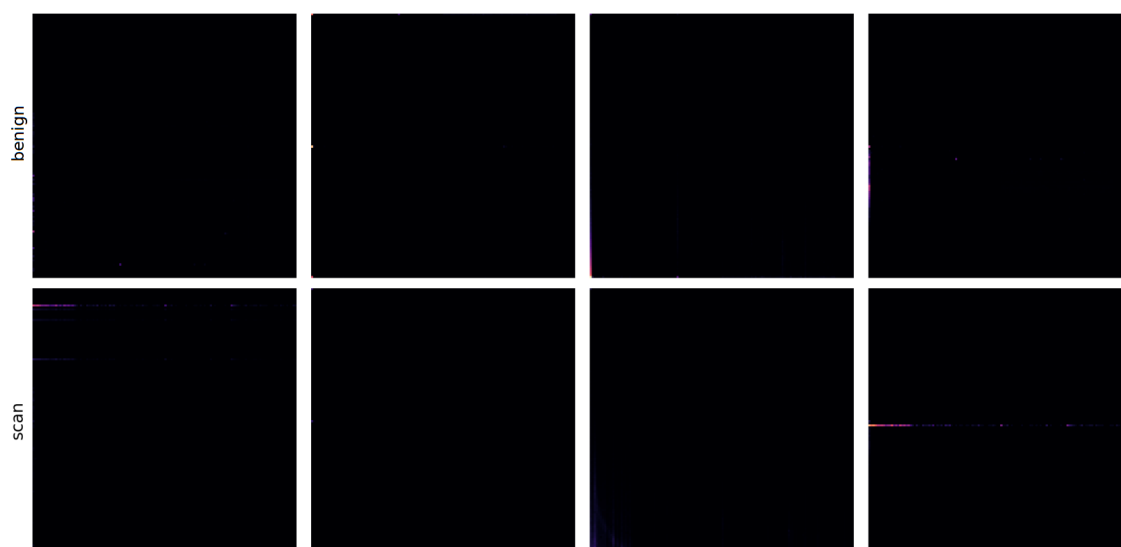


Figure A.11: Mean Channel Images per Class for Port Scanning

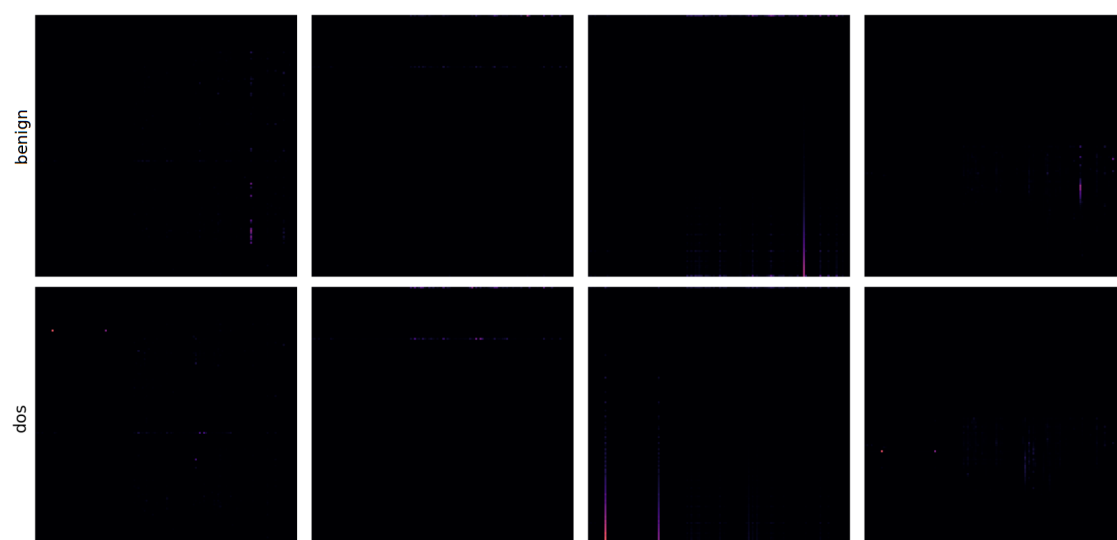


Figure A.12: Mean Channel Images per Class for Denial of Service